FACULTAD DE ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS

DEPARTAMENTO DE ESTADÍSTICA E INVESTIGACIÓN OPERATIVA APLICADAS Y CALIDAD

FINAL DEGREE PROJECT:

A COMPARISON OF MACHINE LEARNING MODELS FOR THE DETECTION OF METASTATIC TISSUE IN AXILLARY LYMPH NODES

Thesis submitted for the Bachelor's Degree in Business Administration and Management in the Universitat Politècnica de València
Academic year 2019-2020

Author: Javier Abad Martínez

Supervised by:
Prof. Dr. Rubén Ruiz García

Valencia, July 2020

Abstract

Breast cancer is currently the leading cause of death due to cancer in women, after lung cancer. For its diagnosis and staging, detection of metastatic tissue in axillary lymph nodes is occasionally used, since lymphatic spread is the main prognostic factor, especially in early stages. However, the pathologist's work in this diagnosis is considerably complex and tedious, so the need to automate this process arises. In the present work, a comparison of several models based on traditional machine learning techniques for the extraction of features and subsequent classification of sections of axillary lymph nodes stained in H&E is developed, to identify whether they contain tissue with metastasis or not. Furthermore, the results are compared with those obtained with other cutting-edge machine learning techniques, specifically with the Convolutional Neural Networks. In this way, alternatives with good enough results are proposed to be applied to the clinical reality of hospitals, with a computational cost and lower needs than the most sophisticated techniques.

Keywords: machine learning, feature extractor, classifier, convolutional neural network, breast cancer diagnosis, axillary lymph node, medical imaging, computer vision

Resumen

El cáncer de mama se impone en la actualidad como la principal causa de muerte por cáncer en mujeres, después del cáncer de pulmón. Para su diagnóstico y estadificación, en ocasiones se recurre a la detección de tejido metastásico en ganglios linfáticos axilares, ya que la diseminación linfática es el principal factor pronóstico, sobre todo en estadíos iniciales. Sin embargo, la labor del patólogo en este diagnóstico es considerablemente compleja y tediosa, por lo que nace la necesidad de automatizar este proceso. En el presente trabajo se desarrolla una comparativa de varios modelos basados en técnicas tradicionales de aprendizaje automático para la extracción de características y posterior clasificación de secciones de ganglios linfáticos axilares tintados en H&E, para identificar si contienen tejido con metástasis o no. Además, los resultados son comparados con los obtenidos con otras técnicas de vanguardia de aprendizaje automático, concretamente con las Redes Neuronales Convolucionales. De este modo, se proponen alternativas con resultados suficientemente buenos para ser aplicados a la realidad clínica de los hospitales, con un coste computacional y unas necesidades inferiores a las técnicas más sofisticadas.

Palabras clave: aprendizaje automático, extractor de características, clasificador, red neuronal convolucional, diagnóstico de cáncer de mama, ganglio linfático axilar, imagen médica, visión artificial

Resum

El càncer de mama s'imposa en l'actualitat com la principal causa de mort per càncer en dones, després del càncer de pulmó. Per al seu diagnòstic i estadificació, a vegades es recorre a la detecció de teixit metastàtic en ganglis limfàtics axil·lars, ja que la disseminació limfàtica és el principal factor pronòstic, sobretot en estadis inicials. No obstant això, la labor del patòleg en aquest diagnòstic és considerablement complexa i tediosa, pel que naix la necessitat d'automatitzar aquest procés. En el present treball es desenvolupa una comparativa de diversos models basats en tècniques tradicionals d'aprenentatge automàtic per a l'extracció de característiques i posterior classificació de seccions de ganglis limfàtics axil·lars tintats en H&E, per a identificar si contenen teixit amb metàstasi o no. A més, els resultats són comparats amb els obtinguts amb altres tècniques d'avantguarda d'aprenentatge automàtic, concretament amb les Xarxes Neuronals Convolucionals. D'aquesta manera, es proposen alternatives amb resultats prou bons per a ser aplicats a la realitat clínica dels hospitals, amb un cost computacional i unes necessitats inferiors a les tècniques més sofisticades.

Paraules clau: aprenentatge automàtic, extracció de característiques, classificador, xarxa neuronal convolucional, diagnòstic de càncer de mama, gangli limfàtic axil·lar, imatge mèdica, visió artificial

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Human intelligence is our ability to reason, solve problems, understand complex ideas, and abstract concepts and learn from our experiences. Intelligence is not limited to an encyclopedic knowledge but includes a deep capacity to understand the environment and to learn from it.

The ambition to create machines capable of emulating human intelligence has been a constant from the mid-twentieth century until today. However, there is a name with which most of the scientific community agrees when it comes to finding the origin: Alan Turing (1912-1954). His well-known publication *Computer Machinery and intelligence* [61] begins the revolution that **artificial intelligence** represents today.

Throughout the 1950s **machine learning** was born as an artificial intelligence application. The principle of it is to design algorithms capable of learning from experience, capable of automating tasks and, in fact, learning from their mistakes and perfecting themselves. Proof of this are the pioneering works of Frank Rosenblatt in the design of the Perceptron in 1958 [48] or the sophisticated models present today, some used in this work.

It is not surprising, therefore, that the applications of machine learning have reached something as sensitive as the medical image. The potential of a machine learning algorithm to learn and to rival with expert pathologists is of particular interest to the scientific community. The medical image is part of a machine learning application known as **Computer Vision**, which aims to design algorithms capable of emulating the behavior of a human eye, being able to classify images based on criteria such as color, shapes or textures. Furthermore, medical images classification is usually a **supervised learning** problem, in which the algorithm is taught with previously classified images, so that it is capable, once it has learned, of classifying unknown (unclassified) images on its own.

Today the most widespread technique in medical imaging are **Convolutional Neural Networks** (CNNs), introduced in 1998 by LeCun et al. in their revolutionary paper *Gradient-based learning applied to document recognition* [36]. CNNs are currently considered as the state-of-the-art for any problem of Computer Vision and, more specifically, of medical imaging. However, CNNs have several disadvantages that are often overlooked and make other more traditional machine learning techniques – such as k-Nearest Neighbors, Support Vector Machines or Random Forest – more convenient in certain circumstances. In particular, CNNs are particularly demanding as far as computing resources go and alternative aforementioned techniques might offer competitive performance at a much reduced computational budget.

1.1 Motivation

This work makes sense in conjunction with another thesis by the same author entitled *Automatic detection of metastatic tissue in lymph node sections of breast cancer patients using Convolutional Neural Networks* [1]. This thesis was presented by Javier Abad Martínez – same author of the present document – as a final project of the degree in Telecommunications Technologies and Services Engineering in July 2020, and was linked to a grant from the Spanish Ministry of Education and Professional Training that same year.

In [1] a model based on CNNs is developed to detect metastatic tissue in sections of axillary lymph nodes removed from patients with breast cancer. Breast cancer is the most common cancer in women and the main cause of death due to cancer after lung cancer. A popular technique to diagnose breast cancer and its stage is by detecting metastases in the axillary lymph nodes, since lymphatic spread is the main prognostic factor.

In this context, both [1] and the present work find their motivation in looking for a model based on machine learning capable of helping pathologists in the diagnosis of breast cancer, specifically in its staging, and being able to define a treatment accordingly. Furthermore, the present document has a second motivation: to compare the CNN model proposals used in [1] with solutions based on traditional machine learning techniques, such is the case of k-Nearest Neighbors, Support Vector Machines and Random Forest. The conclusions can be interesting to start a line of research on which methods are better depending on the circumstances, reaching maximum importance in something as sensitive as the diagnosis of breast cancer.

1.2 Relation with other courses

This thesis also serves as the culmination of the studies of the degree in Business Administration and Management. Therefore, it is relevant to mention the relationship it has with some of the subjects passed throughout the degree. As expected, they are those related to statistics.

Introduction to statistics. Basic knowledge of statistical terminology (e.g. sample, population, mean, variance, etc.) as well as interpretation of descriptive statistics (i.e. graphs, histograms, etc.).

Statistical Methods in Economics. Deepening in statistical concepts and familiarization with the R programming environment. Although the designed models are based on Python 3.7, there is an analogy with R regarding the handling of variables, construction of vectors, methods, functions, etc.

Econometrics. Study of more complex statistical concepts: linear regressions, non-linear regressions, multivariate analysis. Machine learning is based on linear and non-linear models, so it is essential to know its theoretical foundations and applications.

1.3 Summary of the methodology followed

For the preparation of this work, a methodology was followed that occupied most of the academic year 2019-2020, that is, from September 2019 to July 2020.

The methodology followed can be summarized in four parts. First, an introduction to artificial intelligence, machine learning, and the Python programming language. Next, it was necessary to delve into machine learning techniques specialized in medical imaging, as well as to become familiar with the biological background and to do a bibliographic review of how similar problems had been approached. Once the problem and its possible solutions were known, in a third phase different solutions were proposed, mixing techniques and analyzing them with metrics. Finally, this document was written and revised.

In short, the four parts of the methodology followed are:

1. Preliminary work:
    - Introduction to artificial intelligence and machine learning. Specialized bibliography was succinctly reviewed.
    - Introduction to the Python programming language. Through courses from the Coursera platform, the author became familiar with the handling of variables and structures in Python, for example, the generation of loops, conditionals or methods. This was decisive, since all the proposed models are based on this language. It has to be noted that Python programming was not given to the author during his degree.
2. Methodological development:
    - Study of the biological background of the problem. The author studied about breast cancer and its classification based on lymphatic spread. In addition, the importance of the axillary lymph node in this diagnosis was studied in depth.
    - In-depth bibliographic review of the state-of-the-art in the classification of medical images. Several models were studied, which combined feature extractors and classifiers in various ways to face problems similar to that of the present work.
    - Completion of specialized courses in image classification using Python. Specifically, a 40-hour classroom course was attended organized by the Computer Vision and Behavior Analysis Lab (CVBLab) on the use of Convolutional Neural Networks in Computer Vision. CVBLab is a research group that belongs to the Universidad Politécnica de Valencia specialized in signal, image and video processing, as well as in data science and the creation of automatic prediction models.
3. Proposal and selection of models:
    - Characterization of the database. The origin of the database was studied. In addition, its distribution and characteristics were analyzed in order to choose a suitable model.
    - Study of the feasibility of the models. Since some of the models were overly sophisticated or the required materials were not available, an initial filter had to be made.

- Selection of the final models. By trial and error, the models that performed best for this particular problem were selected. Approximately 60% of the total models are presented in this document. 40% were discarded for not achieving the desired results or for being excessively complex or not very intuitive, their explanation being outside the scope of this work.
- Design of the final system. Fifteen model proposals were selected, combining feature extractors and classifiers. The training, validation and test conditions were also defined.
- Study of specialized metrics in machine learning and their use in comparing the proposed models. These metrics were used in the validation and in the test of the models.

4. Writing and revision of the thesis.

1.4 Structure of the thesis

This section summarizes the structure of the thesis. This is also reflected in the content index.

Chapter 1 serves as an introduction to the work, defining some of the concepts that will appear recurrently in the document, as well as a declaration of intent, exposing the motivation of the work and some considerations, such as the methodology followed.

Chapter 2 summarizes the objectives of the present work. General objectives are separated from those related to sustainable development.

Chapter 3 is a review of the state-of-the-art in the classification of medical images. Therefore, supported by several references, the chapter succinctly comments on the avant-garde techniques in this discipline, which serve as examples for the present work.

Chapter 4 studies the biological background of the problem. Therefore, it defines breast cancer, how it is classified and the importance of its diagnosis. Next, the relevance of detection techniques for lymph node metastases to treat breast cancer is highlighted, as is the one in this study.

Chapter 5 develops in depth the theoretical framework of the proposed models. Therefore, it defines their two parts: feature extractor and classifier. The chapter then describes each of these, emphasizing the techniques on which the proposed system is based. Next, it explains the operation of the system, that is, its training, validation and test. Finally, there is a section dedicated to the use of machine learning in Computer Vision.

Chapter 6 defines the methodology followed. Firstly, the section explains the database used, the materials and the metrics with which the models are evaluated. All the proposed models are then explained in detail.

Chapter 7 summarizes the results obtained. To do so, it uses metrics and graphs. It separates the validation and test results.

Chapter 8 covers the discussion of the results. Therefore, it analyzes the values obtained in Chapter 7, first in the validation and then in the test. In addition, there is a specific section where it compares the performance of the system proposed in this document with that proposed in [1], based on CNNs.

Chapter 9 explains the limitations of the work, that is, the biases, assumptions, and simplifications.

Chapter 10 summarizes the conclusions. It serves, therefore, to respond to the objectives stated in Chapter 2.

Finally, Chapter 11 describes the potential future work to improve the present model, in addition to some recommendations following the conclusions in Chapter 10.

Chapter 2

Objectives

2.1 General objectives

This section presents the general objectives of the work. These serve as a guide and are directly related to the conclusions set forth in Chapter 10. The objectives are as follows:

- Study the importance of lymphatic spread as the main prognostic factor in the diagnosis and staging of breast cancer, especially in early stages. Along these same lines, understand the biological background and the usefulness of detecting metastatic tissue in axillary lymph nodes for this diagnosis.
- Develop a model based on traditional machine learning techniques capable of detecting metastatic tissue from sections of axillary lymph nodes stained with H&E (haematoxylin-eosin).
- Create a benchmark to compare various models that combine different feature extractors and classifiers. Therefore, define the specific characteristics of the model or models that obtain the best results and justify them.
- Compare traditional machine learning techniques with other state-of-the-art techniques, specifically with CNNs, in medical imaging applications. Define the advantages and disadvantages of each and the suitability of each solution depending on the problem and the context.

2.2 Sustainable development objectives

Furthermore, it is relevant to highlight objectives directly related to sustainable development. Since the present project aspires to have an industrial application, it is necessary to establish objectives that favor the environment, the economy, and the society. The objectives are:

- Favor the diagnosis and treatment of breast cancer, facilitating these functions for the pathologist thanks to the direct integration of the techniques presented in this document in hospitals.
- Analyze the computational cost and, consequently, the carbon footprint linked to machine learning techniques. Assess and compare different models that can benefit from GPU acceleration to reduce this cost and its environmental impact.
- Democratize the technology developed in this thesis and stimulate the creation of projects along the same line of research, with direct implementation and the ability to help in contexts and problems as sensitive as the diagnosis of breast cancer.

Chapter 3

State-of-the-art review

The application of machine learning techniques to Computer Vision has brought about a real revolution in object detection, classification and segmentation problems. Therefore, it is not surprising that machine learning models have been applied to extremely sensitive fields, such is the case of medical imaging. Medical image comprises the set of procedures used to obtain clinically meaningful information in order to define early diagnoses and prognoses in patients. In this context, an effort has been made in recent years to integrate the human knowledge of medical experts with machine learning techniques. This symbiosis has achieved state-of-the-art performance in the field of medical imaging [13].

The use of machine learning techniques in medical imaging is recurrent in the literature. On the one hand, for more than a decade, some traditional techniques – based on widely known feature extractors and classifiers – have achieved excellent results in problems of this type. Specifically, the k-Nearest Neighbors (k-NN) and Support Vector Machines (SVM) classifiers have traditionally demonstrated a more effective application [68]. Examples of these are, for example, the use of SVM in the diagnosis of prostate cancer in images stained in H&E [67]; of k-NN in histopathological images of breast cancer in this same stain [56]; of SVM and k-NN in magnetic resonance imaging (MRI), combined with feature extractors such as Wavelet Transform and reducing dimensionality using Principal Component Analysis (PCA) [70]; or from SVM in mammograms to which Histograms of Oriented Gradients (HOG) have previously been applied to extract texture features [10]. Furthermore, traditional machine learning is not limited to classifiers such as k-NN and SVM, but there are other techniques that show results with potential applications to medical imaging, such as Random Forest [15] [69], linear Gaussian classifier [60], or some more sophisticated ones like XGBoost [55].

More recently, thanks to the increasing computational power with GPU acceleration and the availability of huge databases, deep learning [35] has become the default machine learning technique for medical imaging applications. Deep learning greatly simplifies feature extraction and pattern identification, making it the solution to virtually any Computer Vision problem. Among all the methods, Convolutional Neural Networks (CNN) have become the most popular. They are also of special interest when applying transfer learning or fine-tuning techniques, where architectures previously trained in ImageNet competitions are exploited [33]. The applications of CNNs are abundant, presenting state-of-the-art results [52][44]. Furthermore, the implementation of more sophisticated methods such as the rectifier linear unit [20] and deep residual learning [24] aims to keep deep learning in its hegemonic position.

However, it is relevant to underline that CNNs cannot be considered the panacea of all medical imaging problems. There are several examples where traditional methods such as those previously exposed obtain better results than deep learning techniques [43]. In addition, there are several criticisms that CNNs receive, based on their high computational cost, their tendency to overfit and their instability, which cast doubt on their real usefulness [3].

Chapter 4

Biological background

4.1 Breast cancer

Breast cancer originates from the uncontrolled proliferation of healthy breast cells, which begin to mutate and form a conglomerate of cells known as a tumor. If the tumor is cancerous, it will be a malignant tumor. Malignant tumors can grow and spread throughout the body, unlike benign tumors, which can grow, but not spread. The spread of cancer through the body through the blood or lymph vessels is called metastasis [6].

The diagnosis of breast cancer in women is more common than that of any other type of cancer, including skin cancer. In addition, it is the second most common reason for death from cancer in women, after lung cancer. According to the American Cancer Society, it is estimated that in 2020 more than 276,480 women will be diagnosed with invasive breast cancer in the United States and will produce a total of 42,170 deaths of women in this same country. Currently, there are more than 3 million women who have been diagnosed with this disease throughout the United States. Finally, the average risk that a woman will develop cancer in the United States throughout her life is around 13%, that is, 1 in 8 women will develop this disease [8].

There are several ways to classify breast cancer. In this work, the so-called staging is the used method, particularly the TNM staging system. TNM is based on the size of the tumor (T); if it has spread through the lymph nodes (N) and, if so, where and how much; and if it has spread to other parts of the body (M), that is, if it has metastasized. Under the TNM indicator, the patient is identified with one of the five different stages, which cover the development of the cancer from the moment the cell becomes carcinogenic (stage 0) to an advanced stage or metastasis where the cancer has spread throughout the rest of the body (stage 4) [6].

4.2 Lymphatic dissemination in breast cancer

The "N" in the TNM indicator corresponds to lymphatic spread. It is relevant to differentiate here between regional and distant lymph nodes. Regionals can be located under the arm (known as axillary lymph nodes), above and below the clavicle, or below the sternum (known as internal mammary lymph nodes). Rather, distant lymph nodes will be located in the rest of the body. Therefore, it is expected that a breast cancer in a lower stage will be present only through regional lymph nodes, reaching the distant ones as the disease progresses.

The spread of cancer by lymph nodes is the most relevant prognostic factor in the initial stages, above the size of the tumor (T) and whether it metastasizes (M). Thus, it is a definitive indicator when diagnosing, classifying, and defining the treatment of a patient with breast cancer [6].

Finally, there is a popular form of diagnosis of breast cancer called sentinel lymph node biopsy (SLNB). The sentinel lymph node is the one to which cancer cells are most likely to spread from the primary tumor. In the case of breast cancer, this is usually the axillary lymph node. Therefore, the procedure will consist on the removal and examination of the sentinel axillary lymph node (SLN) to see if it has metastatic tissue. This study would define whether the cancer has been able to spread throughout the rest of the body, which allows a diagnosis of the patient's stage. Moreover, on examination of the lymph node, the pathologist will require staining to obtain an overview of the tissue samples [32].

4.3 The haematoxylin and eosin (H&E) stain

A popular stain is haematoxylin-eosin (H&E). Since most cells do not have their own color, their direct observation by light microscopy does not allow their morphology to be analyzed properly. To be able to observe it, stains are used in order to color different structures in specific ways, making it easier to differentiate them within a tissue.

Specifically, the haematoxylin-eosin stain colours with haematoxylin the acidic structures in blue and purple while using eosin for basic components, colouring them in pink tones [26]. Figure 4.1 shows some histological images stained in H&E.

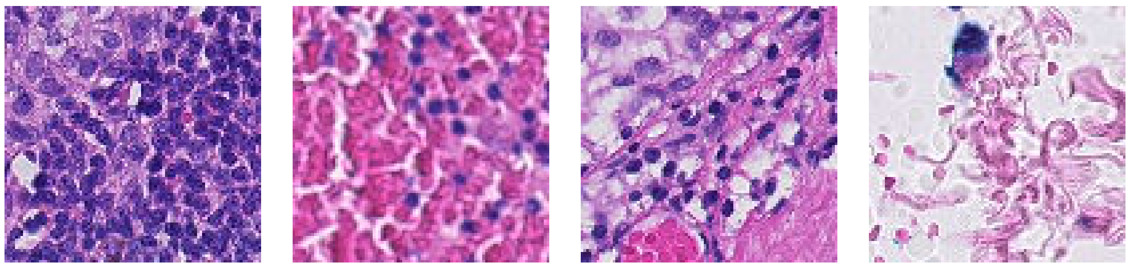


Figure 4.1: Histological images of axillary lymph nodes stained with H&E.

4.4 Automatic detection of metastasis in sentinel axillary lymph nodes (SLN)

New technologies in medicine have allowed high resolution digitization of stains with histopathological tissue, including haematoxylin-eosin staining. Advances in scanners allow the storage of large databases of medical images that can be processed by machine learning models for classification [21].

Chapter 3 studies the current state-of-the-art techniques used for medical images. In the present work, the problem is characterized to the detection of metastases in sentinel axillary lymph nodes (SLN). As mentioned, this analysis has a fundamental relevance in the staging of breast cancer and, therefore, in the definition of a treatment for the patient, especially in early stages.

Applying machine learning to this particular problem is highly desirable. Pathologists' general diagnosis of SLN is especially complex, often leading to erroneous conclusions. One study found that expert pathologists change up to 24% of SLN diagnoses after reviewing them [66]. Furthermore, the analysis is considerably tedious and time consuming. It is, therefore, especially interesting to create an algorithm capable of classifying quickly and with high accuracy a SLN susceptible to metastasize.

Chapter 5

Theoretical framework

5.1 Feature extraction

In any image classification problem, the raw pixel set that conforms the images is a source of excessive information. It would be naive to think that a classifier – however sophisticated might be – is capable of learning from images and identifying patterns as if it were a human eye, being able to recognize all colors, shapes and textures in order to define which category the image belongs to. Therefore, it will be necessary to eliminate the redundant or useless information existing in each of the images that are received as inputs of the system, that is, be able to extract information related only to those parameters that may be relevant so that the classifier can subsequently make a decision based on meaningful data.

For this reason, a feature extraction process is necessary in any machine learning model. The implementation of this previous phase allows reducing the input images into **feature vectors** that are, ideally, informative and non-redundant so that, in short, their interpretation and subsequent generalization to independent datasets are facilitated. Thus, another advantage of feature extraction is the decrease in computational cost and memory usage as classification algorithms and methods only need to work with feature vectors and not complete images.

The process of identifying the features to extract is known as feature selection [2]. In this section some of the most popular algorithms for color extraction (color histograms), shape (Hu moments and Histogram of Oriented Gradients) and textures (Haralick texture) are briefly discussed.

It is important to highlight that, in the present document, dimensional reduction techniques, whose application is widely extended, have been excluded, such as the Independent Component Analysis (ICA) or Two-Dimensional Principal Component Analysis (2DPCA). This decision is due to the fact that it has been considered more didactic and intuitive to use those algorithms whose results are more visual (i.e. extraction of colors, shapes and textures) despite the fact that the efficacy of ICA [39] and 2DPCA [27] is widely contrasted in image classification problems.

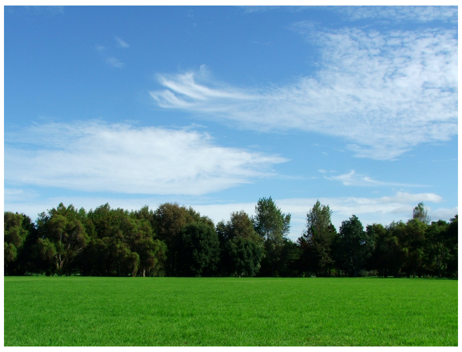
5.1.1 Color histograms

Traditionally, classification problems have been based on text labels, that is, problems where classifiers are capable of defining the category to which each object belongs based on quantitative and qualitative variables such as age, nationality, income, etc. With the development of Computer Vision, the need arises to encode the information of the images using content-based labels, capable of synthesizing information related, for example, to its color.

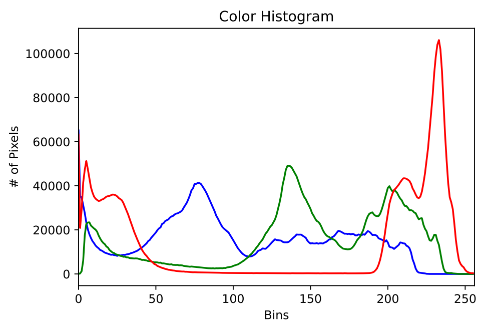
Color in particular is the most intuitive source of information of an image to be able to compare it with another. Many algorithms have been developed since the late 1980s based on color extraction from images [54]. However, one of the most popular techniques is that based on color histograms [28]. In their paper, Jain and Vailayan propose to differentiate each image by means of color histograms, understanding that each image is represented by a set of pixels with values that correspond to colors visible to the human eye. These histograms represent the proportion of each color in each of the images, so that two images will be similar if their histograms – that is, the proportions of each color – are similar. In other words: images with a similar color distribution are semantically similar. Consequently, classifier will assign a class to an image depending on the similarities of the histogram of the image and those of the different classes.

In the Jain and Vailayan approach, three different histograms are computed, one for each color component of the pixels encoded in RGB (Red, Green, Blue). It is relevant to remind that in the RGB color space each pixel quantifies its color intensity with a value between 0 and 255 (equivalent to 8 bits) for each of the colors: red, green and blue. For each of the three histograms, a number of "bins" is assigned for the x axis, so that if 2 "bins" are chosen for the blue color, the value of the blue component of each of the pixels of the image can take two different values depending on the interval under which it falls: [0, 128) or [128, 255].

This way, by examining the histogram of an image, one can have an intuition of the contrast, brightness or color intensity of it. Below is an example of a histogram for the image of a landscape, where 256 "bins" have been used for each color (Red, Green and Blue) and have been plotted in the same graph (Fig. 5.1).



(a)



(b)

Figure 5.1: (a) Original image and (b) Color histogram with 256 bins of the original image in RGB color space.

Since 256 "bins" have been chosen, the intensity of all color combinations is represented. It is easy to see that the predominant color in intensity is blue, which is present in the sky. Regarding the rest of the colors, they are mostly based on the combination of green and red.

However, one of the main problems with the color histograms raised in [28] is that the color space is very sensitive to noise, for example, derived from changes in the illumination. In [31] Jeong proposes the use of the HSV (Hue, Saturation, Value) color space as a solution. This approach is, in fact, the one used in the present document. The concept is the same as before, changing exclusively to the HSV space (see [38] for more information about this color space).

Color histograms still have some drawbacks, as they ignore some features like texture or shape. However, they are a widely used color feature extractor due to their simplicity and the low computational cost involved.

5.1.2 Hu moments

In the field of object detection, the use of moments to describe images has been extensively demonstrated [46]. This implementation is based on modelling an image from a few moments that contain relevant information about it, so that the classifier is able to identify and learn the patterns that define each of the categories. The applicability of moments to describe images is due to the fact that they are invariant, that is, their value is independent, for example, of the orientation or scale used. In this document the use of a group of moments known as Hu moments is briefly explored.

Hu moments were introduced by Hu in his 1961 publication [25]. These invariant moments are calculated from nonlinear combinations of regular moments. The mathematical development of these functions is considerably complex and beyond the scope of this work, however, the author's original document can be consulted in [25].

The success of Hu moments in image classification – and, more specifically, in object detection – is due to its application as a shape feature extractor. Hu explains in [25] how the calculation of the Hu moments provides information about the shapes of an image. Furthermore, due to its invariant condition, this information is insensitive to changes in the rotation, scale, and translation of the images. In this way, for instance, a system that counts the number of people crossing a street throughout the day would not discern between the size of the person (scale) or if, for example, some photographs had been taken upside down (orientation). It is easy to find research works based on these principles, such as those of Paschalakis and Lee in [41] or those of Reeves et al. in [47].

5.1.3 Histograms of Oriented Gradients

Hu's invariant moments are not the only descriptors used in the extraction of shape features. In a more intuitive and visual way, Dalal and Triggs introduced the Histogram of Oriented Gradients (HOG) in 2005. In their publication [14], the authors evaluated several shape feature extractors used to date, and later proposed HOG as the best alternative to these, specifically for pedestrian detection.

The idea of HOG is that the shapes of an image can be described from the distribution that the intensity gradients or edge directions follow, understanding this gradient as the variations in the ordinate and abscissa axes. As with Hu moments, mathematical development and justification are complex and beyond the scope of the present work. The interested reader is directed to [14] for an in depth discussion of the algorithm behind HOG.

The procedure consists of calculating the directional gradients of blocks or local cells in which the image is divided. This is achieved by applying different masks, described in [14]. Once the gradients are obtained, their histograms are computed for each of the blocks. In this way, the HOGs are obtained, which give information about the shape of each of these cells in which the image was divided. An intuitive way to understand it is by representing the histograms using arrows with different length (intensity) and orientation. Lastly, Dalal and Triggs propose a subsequent normalization to avoid the effects of illumination and contrast, since it is an extractor exclusively of shape features and cannot provide information about the color.

Fig. 5.2 is an example of HOG application as shape feature extractors for an image of a dog. The original image, its division into local cells and, finally, how it would be represented by gradient histograms are observed.

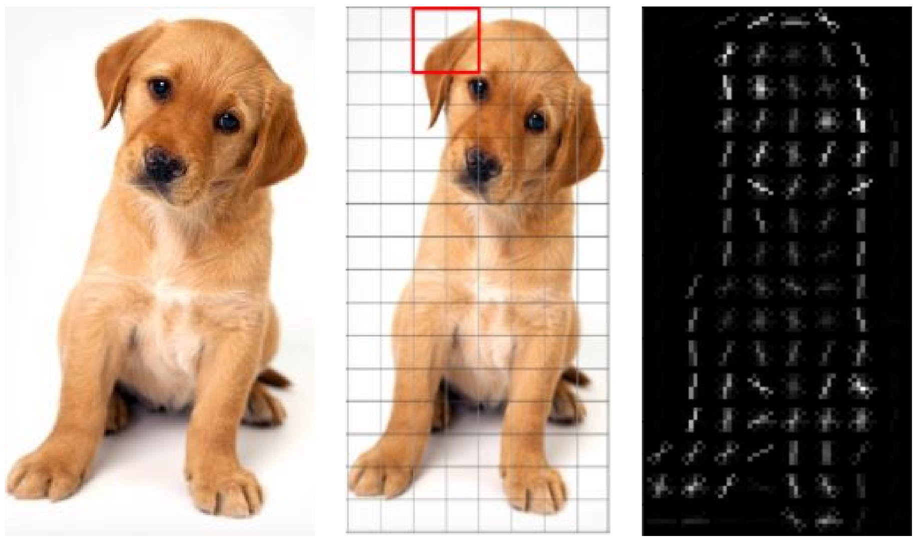

(a) (b) (c)

Figure 5.2: (a) Original image (b) Division of the original image into cells of 8x8 pixels and (c) Representation of the Histograms of Oriented Gradients. Retrieved from *Feature Engineering for Images: A Valuable Introduction to the HOG Feature Descriptor*, by A. Singh, 2019, `https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/`. Copyright 2013-2020 by Analytics Vidhyar.

The histograms have been calculated in local cells of 8x8 pixels, represented in the second image. In the original algorithm there are other modifiable hyperparameters to obtain different HOGs, however, the ones proposed in [14] are normally kept as recommended.

The use of HOGs is common and reproduces excellent results, especially in problems of people detection. Along these lines are the works by Dollár et al. in [17] or by Felzenszwalb et al. in [19], which show the presence of this feature extractor in different predictive models.

5.1.4 Haralick texture

Along with color and shape, texture is another of the most commonly used feature vectors in image classification. Texture is understood as the consistency of patterns and colors in an image, therefore textures help identifying and differentiating between rough and smooth objects or furry and non-furry animals, to name a few examples.

The Haralick texture was defined by Haralick in 1973 in his well-known paper [22]. The fundamental principle of this resides in what is known as the Grey Level Co-occurrence Matrix (GLCM). GLCM collects the count of possible combinations of adjacent pixels in grayscale. It works in grayscale since the color is not relevant and it is possible to represent any grey value with a number between 0 and 255, while in other spaces such as RGB three values are required (one for each component). An example of GLCM if we only consider the adjacent values in the direction from left to right is the one in Fig. 5.3.

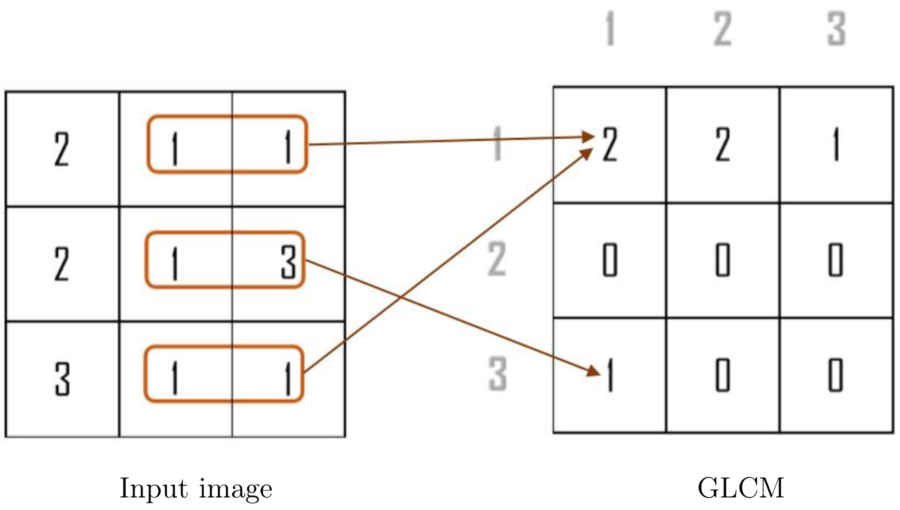


Figure 5.3: Simplified GLCM, only computing the left-right combinations. On the left, the input image. On the right, the GLCM filled with the values of the count of the pairs in the direction X-Y, where the horizontal axis indicates X and the vertical axis indicates Y.

In this way, we observe how the combination of adjacent pixels left-right [1,1] is counted twice, while the combination [1,3] is counted only once. In any case, Fig. 5.3 serves exclusively as an example due to its simplicity, the GLCM used in the Haralick texture would take into account all possible pixel combinations: up-down, down-up, right-left, left-right. A rough surface will therefore have a GLCM matrix that shows sudden and constant changes in grey intensity, while a soft surface will demonstrate the opposite.

The use of the Haralick texture, especially in medical images, is extremely popular. Its first applications were in the analysis of ultrasound images of livers [37] and today it finds utility, for example, in oncology [40] or in magnetic resonance images (MRI) [51].

5.2 Statistical classification

In Chapter 1, supervised learning was introduced as the set of systems and algorithms based on artificial intelligence – and, more specifically, on machine learning – that make up a predictive model that learns from labelled data to later be used to classify unlabelled data. The model must be able to find patterns using an appropriate learning algorithm, such as linear regression or Random Forest.

Furthermore, these models can be divided into two very broad groups, depending on the problem they seek to solve: statistical regression and statistical classification. Regarding the first, the model tries to find solutions represented with real variables. An intuitive example is trying to explain a person's weight from a linear relationship where there is a direct proportionality between, for example, the height and weight of the individual. As simple as this sounds, this model would be a concrete implementation of machine learning known as linear regression. On the other hand, statistical classification is based on assigning a category or class to unclassified data using a model that has learned from correctly classified previous data. That is, in the statistical classification the output is discrete, unlike in the regression.

The model developed in this work belongs to this second type. The importance of the classifier choice is indisputable in any machine learning model: it is here that the theoretical plane finds practical application, which gives real value to the algorithm. The classifier must be able to understand the feature vectors extracted in the feature extractor and decide which category an object belongs to, based on them. This section aims to give basic notions of some of the most widely used classifiers in statistical classification: k-Nearest Neighbors, Support Vector Machines and Random Forest.

5.2.1 K-Nearest Neighbors

References to the method of the nearest-neighbors have been present in the literature for more than 50 years, however, Cover and Hart have been credited as the first to popularize it with their famous paper [12]. The method stands out for its simplicity, not in vain, it could be said that the model does not even "learn". Nearest-neighbors is actually an instance model. This means that the instances learned in the training are used as a "knowledge base" for future predictions, without learning from the predictions latter and, therefore, maintaining the model immutable once it has been trained.

K-Nearest Neighbors (k-NN hereafter) bases its operation on the distance between feature vectors. K-NN assumes that two objects will be more similar the smaller the difference between the feature vectors that describe them. In this way, in its simplest version, it will assign the new object the category of the closest object. Fig. 5.4 serves as a graphic description of this simple model.

In this specific case, the hyperparameter $k = 1$. You could define k as the number of neighbors that are taken into consideration when making the decision about which category a new object belongs to. In this way, k-NN works as a voting system, where the k nearest neighbors are taken into account and the new object is assigned to the category of the majority. Therefore, the decision on k is the most important in systems that use k-NN as a classifier. Fig. 5.5 shows how the previous example would have yielded different results with another value of k.

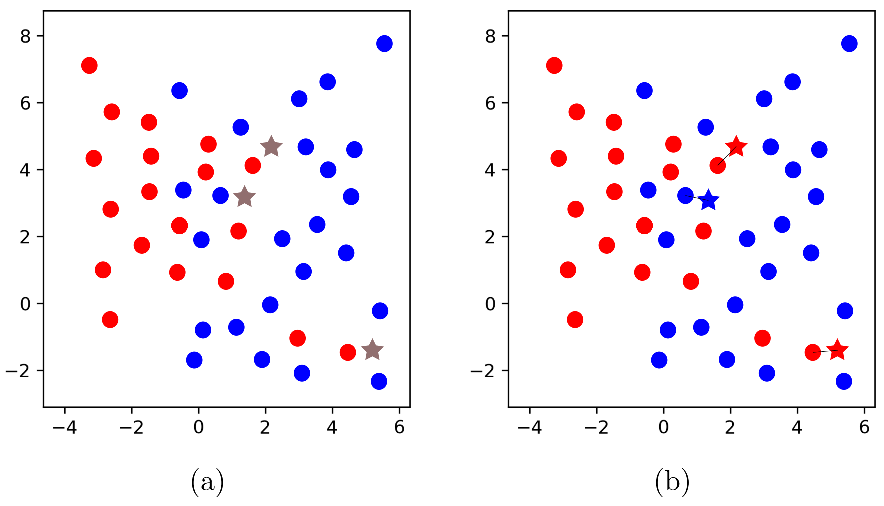


Figure 5.4: (a) Distribution of the training set, the stars represent unclassified objects. (b) Classification of the objects using $k = 1$.

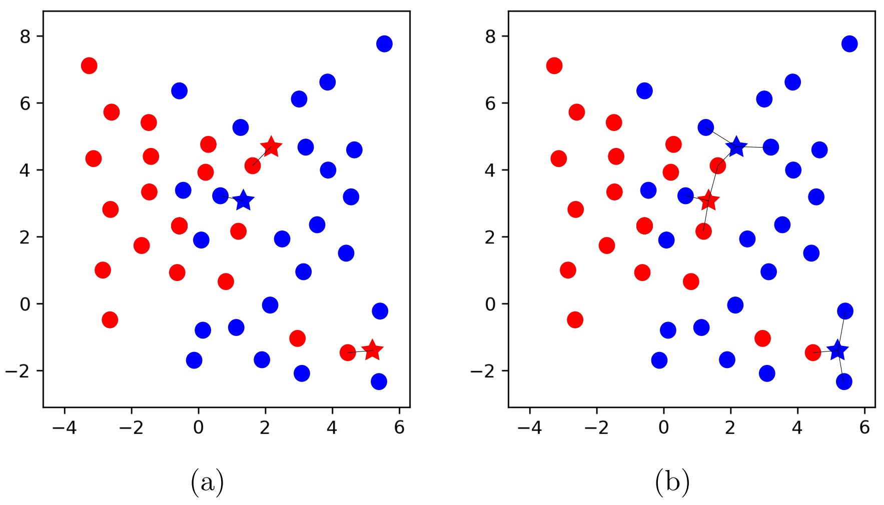


Figure 5.5: (a) Classification with $k = 1$ and (b) Classification with $k = 3$.

However, the value of k is not the only decision to make when programming a k-NN model. It has been mentioned before that two objects will be more similar when the distance between them is smaller. Therefore, it will be necessary to define now how this distance is computed.

Some of the most popular options are Euclidean distance (5.1) and Manhattan distance (5.2). These follow the expressions:

$$d\left(p,q\right) = \sqrt{\sum_{i=1}^{n} \left(p_i - q_i\right)^2} \tag{5.1}$$

$$d\left(p,q\right) = \sum_{i=1}^{n} \left|p_i - q_i\right| \tag{5.2}$$

Other alternatives are Chebyshev distance, Mahalanobis distance or chi-square distance.

The number of k neighbors and the type of distance used are the most important decisions in a k-NN model. However, the evolution of k-NN over the years has allowed the incorporation of new hyperparameters for more sophisticated systems. Some of them are:

- The weights used to compute the distances. Uniformly assigned weights will give equal importance to two neighbors in the vote, regardless of which one is closer. In contrast, a weighted assignment based on distance will give the closest neighbors more value when assigning the category to a new object. The former approach refers to “uniform weights” whereas the latter refers to “distance weights”.
- The algorithm used to define the k nearest neighbors. In more complex problems, it is not enough to compute the distance – whether Euclidean, Manhattan or other – to find the nearest neighbors. The need then arises to implement algorithms such as Ball-Tree or KD-Tree. In [45] you can see a comparison between these, in addition to some of their advantages and disadvantages.

5.2.2 Support Vector Machines

The use of Support Vector Machines (SVM hereafter) as classifiers is ubiquitous in the world of machine learning due to its versatility, low computational cost, and evolution and refinement over the years. Vladimir Vapnik established its foundations while working for the telecommunications company AT&T in his 1995 publication [62]. Unlike k-NN, SVM is based on a complex and unintuitive algorithm. Therefore, in this section the given explanation will be limited to a simple example of SVM in a problem with two categories and two features.

The principle of SVM resides in separating the existing categories by means of one or more hyperplanes, where the objects to be classified are located in an N-dimensional space where N is the number of identified features (e.g. age, height, color, etc.). As mentioned previously, in this explanation the problem is limited to two categories and two features (two-dimensional space), so the hyperplane that separates them will be a single straight line.

Fig. 5.6 shows this example graphically: first the distribution without separating and then the optimal hyperplane. This optimal hyperplane will be the one that maximizes the distance between the element closest to the hyperplane and the hyperplane, known as the **margin**, for each of the categories.

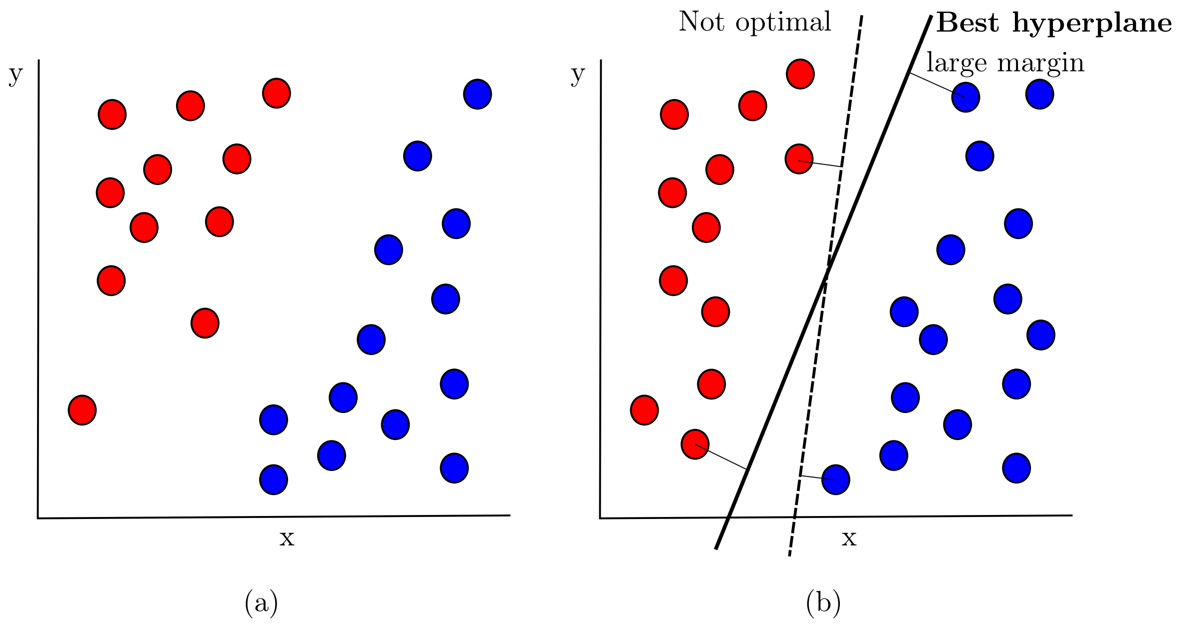


Figure 5.6: (a) Labeled data. (b) In 2D the best hyperplane is a line.

Expressing it mathematically, the hyperplane $g(\vec{x})$ is found such that:

$$g(\vec{x}) = \vec{w^T}\vec{x} + w_0 \tag{5.3}$$

So that:

$g(\vec{x}) > 1, \forall \vec{x} \in class\, 1$
$g(\vec{x}) < 1, \forall \vec{x} \in class\, 2$

Where $\vec{w}$ are the weights to find and w_0 is the independent term of the hyperplane. The value of these weights will be the ones determined by the hyperplane that meets the aforementioned maximum margin condition. Therefore, knowing that the distance z between the point closest to the hyperplane and the hyperplane is:

$$z = \frac{1}{||\vec{w}||} \tag{5.4}$$

The value of $\vec{w}$ should be minimized since it will maximize the distance z.

The procedure for minimizing weights includes the use of Lagrange multipliers and the Karush - Kuhn - Tucker conditions (KKT). The full development can be found in [62].

Also, unlike k-NN, SVM is a learning model. Learning implies that the weights $\vec{w}$ are updated as new data is incorporated, that is, as new objects are classified. Again, to know the details about this process of regularization of the weights, it is recommended to consult the original paper [62].

In the example, it was illustrated how easily the two categories could be separated by means of a straight line. However, in most cases the categories are not so radically differentiated and, therefore, it is impossible to separate the distributions by a straight line. Keeping the example of two categories and two features, Fig. 5.7 shows a more complex dataset, where the separation is not trivial.

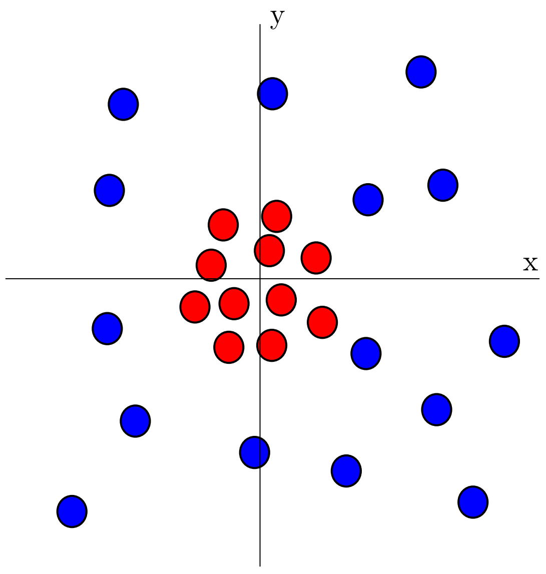


Figure 5.7: A more complex dataset.

A solution to this problem could be the addition of a third dimension, represented by the z axis. This would allow the categories to be separated as shown in Fig. 5.8.

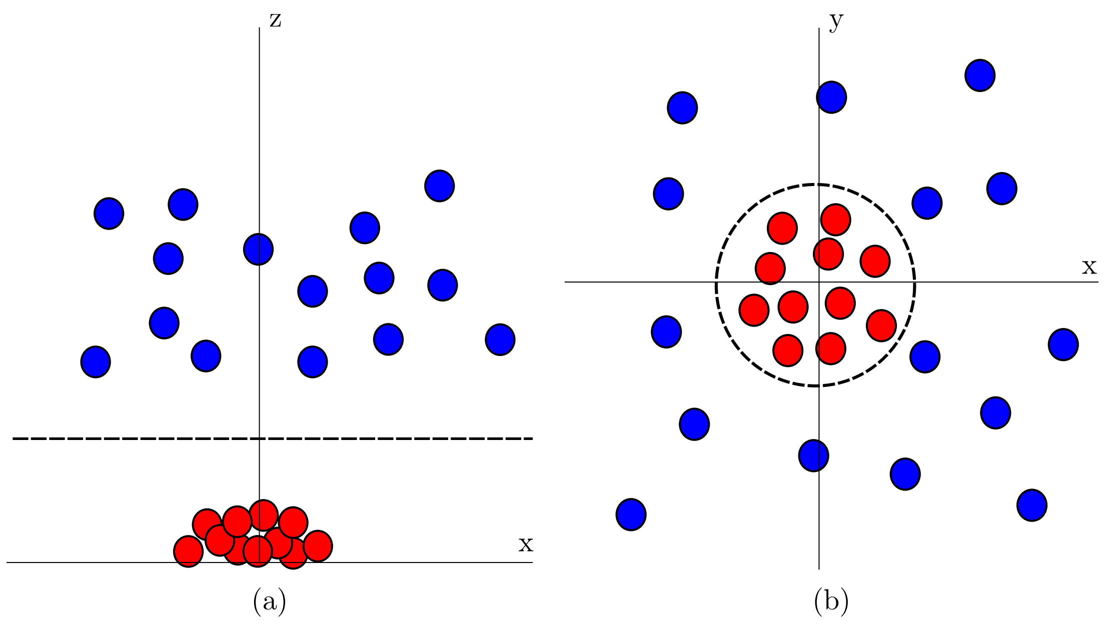


Figure 5.8: (a) Separation from a different perspective using a line and (b) Representation in the original view, with a non-linear separation.

However, adding dimensions is not practical with complex distributions where the classes are not well differentiated and there are several categories that, in short, exponentially increase the computational cost. Vapnik proposes a solution in [62] that is possibly the reason for SVM's success. Vapnik is able to mathematically express any necessary new dimensions using only the variables of the current dimensions. This is accomplished through scalar products – or, as they are referred to in the paper, dot products – that can be modified using linear and non-linear expressions. These modifications of the dot product are known as the kernel trick functions and are not exclusive to SVM but are applicable to any classifier.

As can be seen, the theory behind SVM is rather complex, however its implementation in practice is relatively straightforward. What is important to highlight are the three main points of SVM: the use of the hyperplane to discriminate categories, the optimization of the hyperplane based on the maximum margin and the use of kernel functions to extend solutions from linear to non-linear.

5.2.3 Random Forest

Breiman's proposal in 2001 for the use of Random Forests as a classifier [5] represents a before and after in statistical classification. Random Forest is consolidated as a powerful model capable of predicting large datasets with a small computational cost and very high accuracy.

To understand the concept of Random Forest one must be familiar with Decision Trees. Decision Trees are abundantly present in many disciplines, and they consist on classifying objects based on a sequence of usually binary questions. A simple example can be seen in Fig. 5.9.

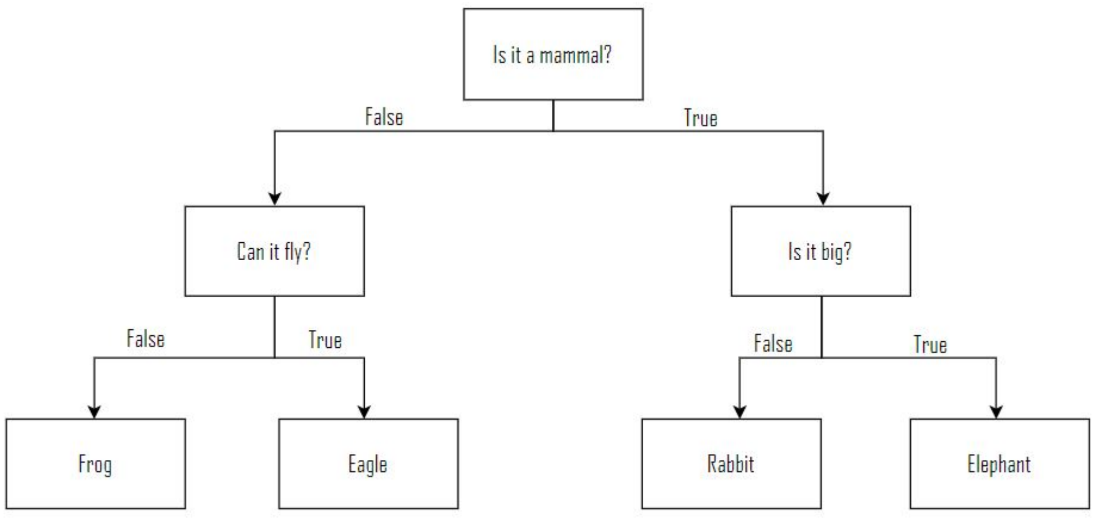


Figure 5.9: Example of a Decision Tree.

It is worth mentioning that, in the case of continuous features, the assignments to each branch will depend on whether the value is included in a defined interval or not.

Random Forest operates as an ensemble of Decision Trees. In statistical classification, an ensemble is a combination of models that provides better performance than each of its members individually. The principle is as follows: Random Forest algorithm will generate several Decision Trees, different from each other, to predict the class to which the object belongs. Once all the predictions have been made, the final class will be the most voted, as was the case with k-NN. In this way, despite the fact that each Decision Tree will include error and bias, they will decrease when averaging with the rest, since other trees may have made very accurate predictions.

However, for this to really work, the errors that each of the trees make must be different from the others, that is, they must have reached the conclusions in different ways. In other words, Decision Trees in Random Forests have to be uncorrelated – or, at least, very little correlated. This is where the algorithm value is found. Specifically, uncorrelatedness is achieved through bagging (Bootstrap Aggregation) and feature randomness.

Decision Trees are very sensitive to the data with which the model training is carried out, therefore, bagging is used to generate different trees. Bagging consists of training the model using samples of the same size as the original dataset, but with replacement. For instance, if the data is [1,2,3,4] a possible sample would be [1,2,2,4] or [1,1,1,1]. Each of these samples is called bootstrap. Bagging is also exemplified in Fig. 5.10.

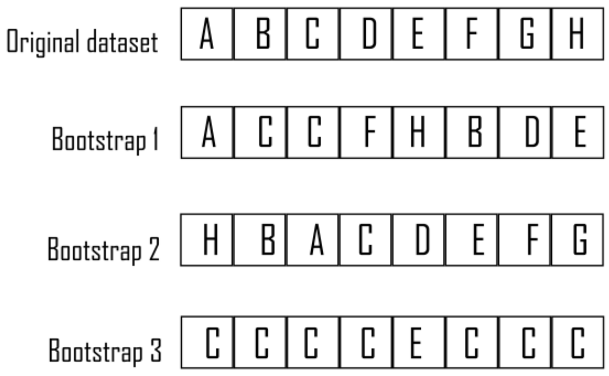


Figure 5.10: Example of bagging.

Feature randomness implies that each of the constructed trees will have a different set of features to classify the objects. For example, if the features used to classify are age, height, and weight, one tree could base its predictions on age and height, while another on height and weight, or even just on weight. The number of features taken into account for each tree must always be less than the total, unless otherwise indicated. Fig. 5.11. serves as an example, where the behavior of an individual Decision Tree is compared to a Random Forest made with two Decision Trees applying feature randomness.

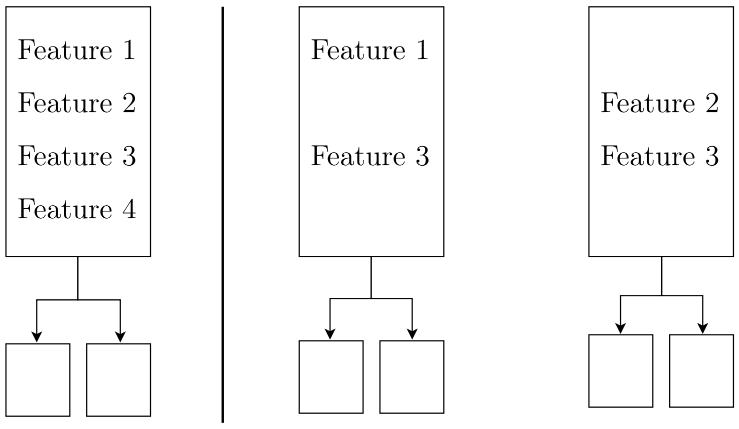


Figure 5.11: On the left, a typical Decision Tree. On the right, two possible trees obtained by feature randomness.

Once defined, it is relevant to know the hyperparameters in order to build a powerful Random Forest. Some of these are listed below.

1. The classification criteria. It is important to define the criteria by which each branch is generated, that is, the "question" that is asked in the Decision Tree node and that leads to two different branches. Fig. 5.9 illustrates questions such as "Is it big?". For continuous quantitative variables, the most popular criteria are the Gini and the Cross-Entropy criteria. Even though they are not complex concepts, they are outside the intention of this document. However, information regarding them can be found in [58].
2. Maximum tree depth.
3. Maximum characteristics in each tree.
4. Number of trees.

In short, Random Forest properties have positioned them as one of the most useful classifiers in supervised learning. They demonstrate obtaining excellent results, in addition to knowing how to handle large databases, while taking into account all the features in the prediction.

5.3 Training, validation and test

The previous sections illustrated how to build a machine learning model, understanding it as a system made up of a feature extractor and a classifier. This section explains how to train, validate and test this model.

There are several approaches that can be found in the literature on how to split the dataset. The traditional approach defines a single partition to split training and test data. The idea is this: if the model has been adjusted – that is, it has learned – from a dataset, it would be unfair to evaluate its performance on that same dataset. Therefore, an independent dataset (test) is reserved to test the performance of the fitted model on an unknown and unbiased dataset. Specifically, in the case of supervised learning, the labels of the test set are known and, therefore, testing serves as an evaluation of the model, comparing the correct labels with the predicted ones (the most common way of evaluating a model is simply dividing the number of correct predictions and the total). This vision is the one that arises, for example, in recognized machine learning books [29] [34].

In this document, however, we bet on a double partition. After dividing the original dataset between training and test, the training set is divided again between "real" training and validation. This validation set allows comparing different models, with different classifiers and hyperparameters (as will be seen in the next section), before evaluating the definitive model in the independent test set. In this way, the use of the test set is limited to the generalization of the final model. This position is also recurrent in the literature [50].

Therefore, from now on this second approach will be maintained to train, validate and generalize the model. It is convenient to summarize the use of each of the identified sets:

- Training set: adjusts the chosen hyperparameters recursively. For example, the hyperplane weights in SVM changed as new data entered the model during training. In general, more training data will lead to better fitted models.

- Validation set: allows one to compare different models with different hyperparameters by evaluating their predictions against the correct labels – what is known as ground truth. The validation, therefore, has applicability in the hyperparameter tuning (see Section 5.4).
- Test set: once one or more models have been chosen, a final evaluation is carried out on another independent data set. This will be the one that ultimately defines which model is better.

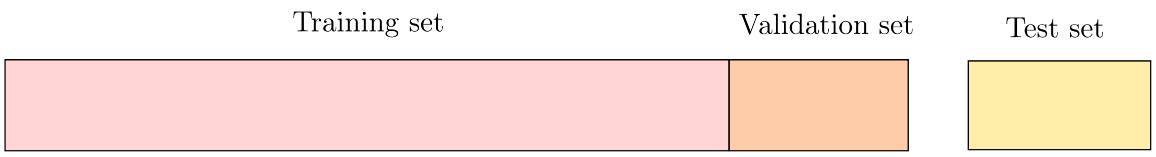


Figure 5.12: Double partition of the dataset: training and validation, and test.

5.3.1 The k-fold cross-validation

The use of an only validation set to obtain unbiased evaluations of the different models is not always enough. If the dataset is separated into a single training set and a validation set, the hyperparameters of the model will be adjusted to this specific training set, in such a way that the evaluation would differ if the partition were different. Furthermore, the model fits exclusively with the training set so that, if you train too much, you run the risk of identifying excessively specific characteristics of that particular dataset, making it impossible to properly generalize to an independent dataset. This problem is called overfitting [7].

To avoid this, k-fold cross-validation is used, introduced in 1974 by Stone in his book [57]. Parameter k defines the number of groups into which the training set is divided, so the training of $k-1$ groups is carried out, leaving a separate one for validation. In this way, each of the k groups is used once for validation and $k-1$ times for training. As a result, k different evaluations are obtained, from which the average is calculated. Once trained and validated, it is evaluated with the test in the same way as before. Fig. 5.13 illustrates a schematic of how cross-validation would be performed with $k = 10$.

Regarding the choice of k, it must be such that each of the divided sets has a representative amount of data. Empirically it has been shown that values of $k = 5$ and $k = 10$ give good results in terms of minimizing bias and variance in each of the validations [29].

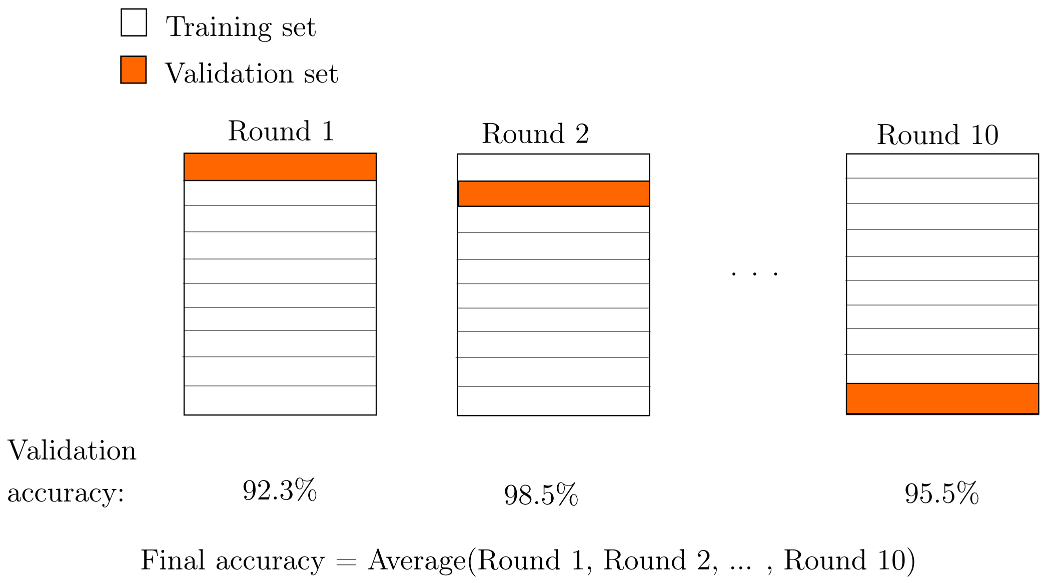


Figure 5.13: K-fold cross-validation with $k = 10$.

5.4 Hyperparameter tuning

The term hyperparameter has been mentioned in different sections. It is now convenient to define exactly what hyperparameters are and how they differ from normal model parameters. Hyperparameter is understood as all those parameters that control the learning process of the model and that can be defined by the programmer. Therefore, the number of neighbors k in k-NN, the kernel (linear or non-linear) used in SVM or the number of trees in Random Forest are examples of hyperparameters. On the other hand, there are model parameters – simply called parameters – that are not modifiable, but are defined during the training itself. Examples of these are the weights in SVM.

It is clear that the setting of the hyperparameters greatly conditions the performance of the model. This is why an optimization of these is used, known as hyperparameter tuning, which is directly related to the cross-validation seen in Section 5.3.1. Simply, a cross-validation of all the desired hyperparameter alternatives is carried out, choosing as definitive the one that better performance shows [11].

If the hyperparameter search is exhaustive, i.e. cross-validation of all alternatives is performed, it is called Grid Search. If, on the other hand, not all the alternatives are tested, but only some of them chosen at random are validated, it is known as Random Search. Random Search is computationally less expensive and its results have been empirically proven to be good enough to replace Grid Search in virtually any problem [4]. In short, the training, validation and testing of the model, integrating both cross-validation and hyperparameter tuning are summarized in Fig. 5.14.

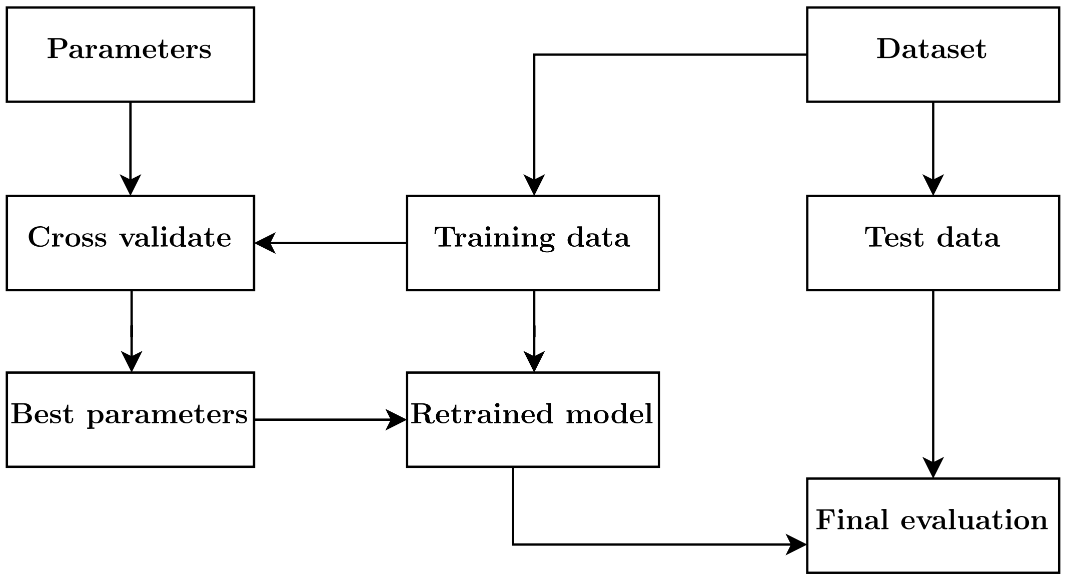


Figure 5.14: Hyperparameter tuning general flowchart.

5.5 Machine learning in Computer Vision

Throughout the previous sections, especially when explaining classifiers, an attempt has been made to illustrate machine learning theory by means of simple examples with a small number of highly intuitive variables (e.g. age, weight, height). However, this is not so trivial in the world of Computer Vision, so it is relevant to comment on the particular application of machine learning in it. Here, the features in many occasions are not quantitative or qualitative variables of a single value, but are vectors made up of many of them. This is what Section 5.1 was referring to with feature vectors. Thus, when computing the color histogram of an image, this feature is not a single value, but rather hundreds or thousands of values that explain the information related to the color distribution of the image.

This appreciation was taken into account when some types of feature extractors were illustrated, but not with the classifiers, where a simple explanation was preferred. The classifiers in Computer Vision calculate different indicators of the feature vectors – such as the mean or variance of their values – and separate them into intervals, rather than into single values. If this is not done, for example, each image would have a different color histogram (no image is identical, however similar it may be) and, therefore, a different feature, generating excessively complex spaces, with thousands of dimensions. This is why, for simplicity, the calculation of means or variances and their grouping into intervals are used.

Despite attempts to simplify the dimensionality of the problem, the use of machine learning in Computer Vision remains considerably complex. It is for this reason that in these applications numerical computing reaches a fundamental relevance: faced with the impossibility of manual methods of simple statistics, it is now the computer that assumes the fundamental role. The role of the programmer will be to facilitate this task for the machine, defining an appropriate model by contrasting different feature extractors, classifiers and hyperparameters and thus defining which one produces the best results.

Chapter 6

Methodology

6.1 Materials, apparatus and procedures

This section describes the materials, apparatus and procedures followed and used in the development of the experimental part of the work. It begins with an illustration of the materials used, that is, the hardware and software needed. Next, the database containing the images is explained, that is, their format and the process that was carried out in order to obtain them, as well as some relevant considerations. Finally, the meaning and calculation of the metrics used to evaluate the project results are justified.

6.1.1 Equipment and programming environment

The project was carried out on a computer with an 8th Generation Intel® Core ^{TM} i7-8750H processor with six cores, with 16 GBytes of DDR4 RAM running at 2666MHz. Moreover, the hardware includes a NVIDIA GeForce GTX 1060 graphics card. The operating system used is Windows 10 64-bit.

We have chosen to implement the different models proposed in Google Colab mounted on Google Drive, to facilitate the transfer of files. Google Colab is a free environment based on the open source project Jupyter Notebook that allows its users to run and implement machine learning models in the cloud. Jupyter is an interactive environment that allows Python code to be executed dynamically, acting as a client-server application. In this way, Google Colab allows users to take advantage of the simplicity of Jupyter using Google Virtual Machines (VMs), to which the computer connects remotely. Google Colab VMs offer both Tensor Processing Units (TPU) and Graphical Processing Units (GPU) as computing resources, however, since the code used is not compatible with GPU-accelerated packages, the use of these is disregarded. In particular, a Google VM CPU is used for the present work, specifically an Intel Xeon processor with two cores @ 2.3 GHz and 26 GBytes of RAM. The following picture, taken directly from Google, pictures a typical Google Cloud infrastructure.

Figure 6.1: Picture of Google's Cloud TPU v3 Pods. Retrieved from *Google Cloud. Cloud TPU*, by Google, `https://cloud.google.com/tpu?hl=es-419`.

The programming language used is Python 3.7, using the high level framework specialized in machine learning scikit-learn, specifically in its version 0.23.1. In addition, other libraries are required such as Numpy (version 1.18.4) for scientific computing, Pandas (version 1.0.3) for data manipulation and analysis, and OpenCV (version 4.1.1) for Computer Vision.

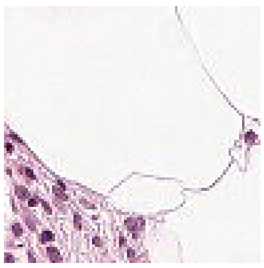
6.1.2 Characterization of the database

The database used comes from a Challenge organized by Kaggle, a subsidiary of Google LLC. Kaggle is a community specialized in machine learning, where experts and amateurs can share knowledge and keep themselves updated with the newest trends. However, Kaggle's main objective is to organize machine learning competitions where users from all over the world compete to solve a problem. The problem that is addressed in this document can be found in [63], and consists of, as explained on several occasions, identifying metastatic tissue in patches of sections of axillary lymph nodes, stained with H&E. The Kaggle database comes from another database known as PatchCamelyon (PCam) [64]. The difference between the two databases is due to the fact that the Kaggle database has eliminated duplicated images that PCam contained. Also, PCam derives from the original database: The Camelyon16 Challenge.

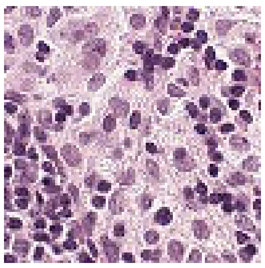
The Camelyon16 Challenge [65] was a Grand Challenge organized in 2016 by the International Symposium on Biomedical Imaging (ISBI) in which research groups from around the world competed to develop a machine learning algorithm that was capable of identifying metastases in axillary lymph nodes of women with breast cancer. Some notable participants were Harvard Medical School or MIT. The paper [18] details the conditions of the competition. However, in this section it is only relevant to mention, without going into far too many details, how the images were obtained. The original Camelyon16 Challenge dataset was made up of 400 WSIs (Whole Slide Images) obtained from patients at Radboud University Medical Center (RUMC) and University Medical Center Utrecht (UMCU). RUMC images were obtained with a digital scanner Pannoramic 250 Flash II from 3DHISTECH, while UMCU images were obtained with the NanoZoomer-XR Digital slide scanner C12000-01, produced by Hamamatsu Photonics. The images were classified by expert pathologists. Furthermore, the categories – cancer and non-cancer – are balanced following the proportion 50/50.

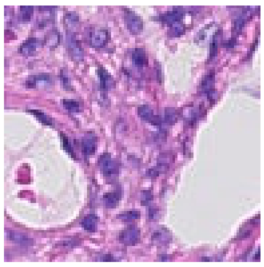
Since the Grand Challenge WSIs are considerably difficult to manage, the PCam is a simplification of it. In PCam the images were converted to the HSV color space, blurred and passed through a filter that eliminated those whose saturation was less than 0.07. Each of the WSI was sampled obtaining several 96X96 pixel patches from each WSI, and saved in .H5 format. Each patch is marked with an annotation that takes the value of "1" if the central region of size 32x32 pixels contains metastatic tissue and "0" otherwise. The area outside the 32x32 section does not influence the annotation, it is simply used to facilitate the use of some models that do not apply zero-padding (adding zeros as a contour of the images). The sampling was done by iteratively choosing a WSI and selecting a patch within it, that could have a positive (cancer) or negative (non-cancer) annotation with a probability of p. For consistency with the original dataset, the p was selected such that the ratio of positives to negatives was 50/50. The annotations for each image are saved in a .csv file identified with the image to which they refer. The final dataset consisted of 262,144 images for training, 32,768 for validation and 32,768 for test.

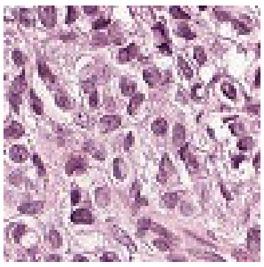
The Kaggle challenge, in short, further simplifies PCam. The first difference is that, as mentioned, it eliminates the duplications that appeared by the probabilistic sampling in PCam. In addition, the format of the images is .tif and not .H5, therefore, it facilitates even more their handling. Regarding the rest, it is identical to the PCam database: image size, annotation system, etc. Kaggle also facilitates the split between images: 220,025 for both training and validation and 57,458 for test. However, the whole test set available in Kaggle was not labelled and only the 220,025 set was left available for all training, validation and test. Fig. 6.2 shows some of these images with their respective annotations.


(a) Class "0"

(b) Class "1"


(c) Class "0"


(d) Class "0"


(e) Class "1"

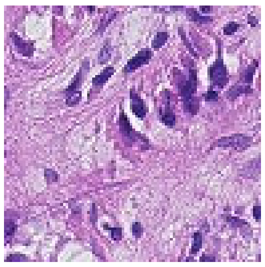

(f) Class "1"

Figure 6.2: Six randomly chosen images from the training set with their respective labels. Class "1" refers to tissue that metastasizes, while Class "0" refers to healthy tissue. Retrieved from *Histopathologic Cancer Detection Dataset*, by B. Veeling, `https://www.kaggle.com/c/histopathologic-cancer-detection/data`. Copyright by Kaggle

From this database, however, in this study only 50,605 images are used, that is, approximately 23% of the available dataset. The reason is as follows: the Kaggle dataset has a very high number of images since it is aimed at models based on Convolutional Neural Networks, which have a much higher computational cost and can be accelerated with GPU, unlike the models described in this document that can only use CPU.

The 50,605 images were chosen at random to avoid bias, taking 23% of the total since it empirically generates reasonable times and results. A partition of 44,005 images was made for both training and validation and 6,600 for test, maintaining the recommended proportion of between 10% and 15%. Fig. 6.3 shows the distribution of the training and test sets in terms of their proportions of positive (cancer) and negative (non-cancer) annotations.

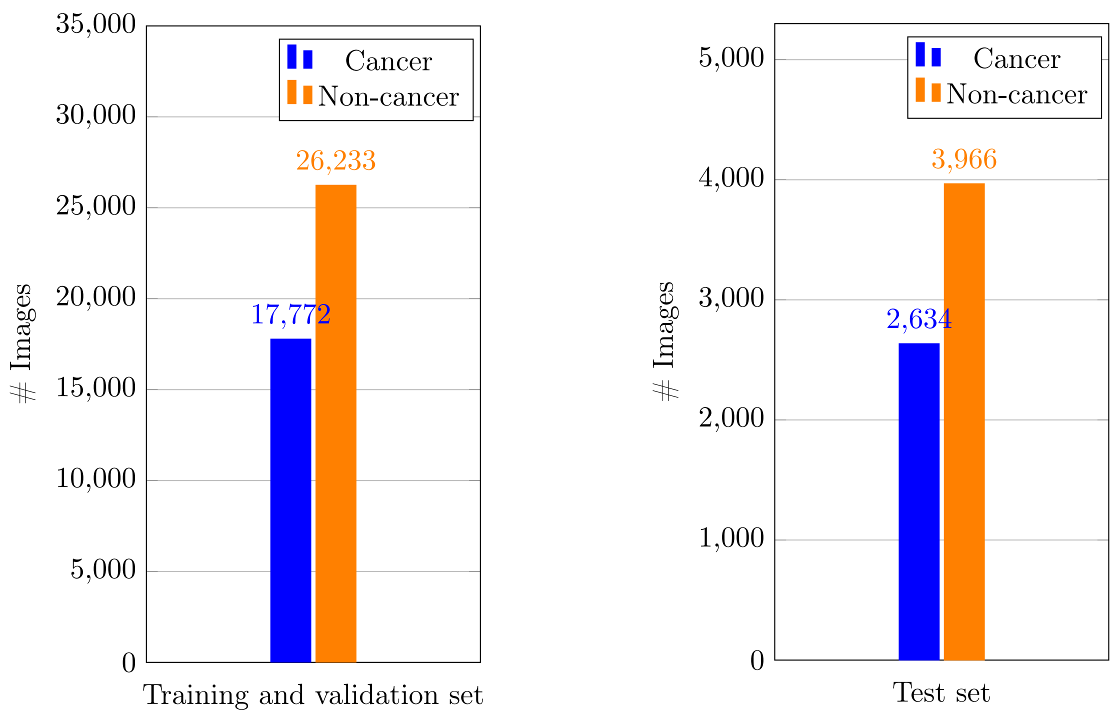


Figure 6.3: Distribution of the classes (cancer and non-cancer) in the training and validation set and the test set.

6.1.3 Description of the metrics used

Chapter 7 presents the results of the predictive model, both for validation and for test. Therefore, it is important to define the metrics that will quantify the degree of success of each of the models.

First, the following simple concepts can be defined as they will serve as indicators within the metrics themselves. These are the types of solutions that each classification can lead to, and they are as follows:

True positive (TP): the model detects metastatic tissue in the section and, indeed, the lymph node presents it.

False positive (FP): the model detects metastatic tissue in the section, but the lymph node has only healthy cells.

True negative (TN): the model does not detect metastatic tissue and, indeed, the lymph node presents only healthy cells.

False negative (FN): the model does not detect metastatic tissue in the section, but the lymph node does present it.

These indicators can be summarized in what is known as the confusion matrix, which is exemplified showing the four alternatives graphically in Fig. 6.4.

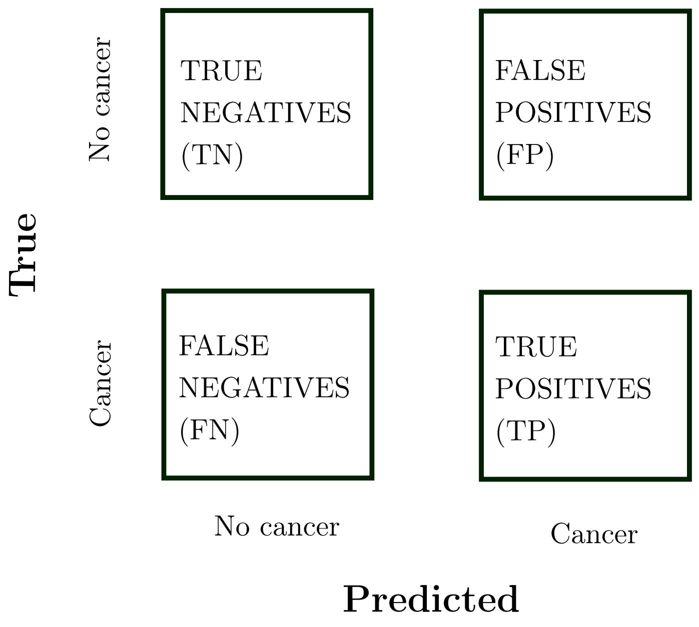


Figure 6.4: Confusion matrix with the possible solutions in our problem.

Furthermore, using these indicators, more complex and informative metrics can be formulated [30]:

Accuracy. It is the most intuitive metric and the one generally used in validation. It is calculated using the quotient between the correctly classified categories and the totals. Its main limitation is the lack of information: it is an excessively generic metric and does not give information on how powerful the algorithm is to identify metastasis or healthy tissue. Obviously, the higher this value, the better, reaching the unit if all the predictions have been correct.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6.1}$$

Precision. Defines the power of the model to correctly classify a particular category. In the present work, the main interest is to detect in which patches metastatic tissue exists. Therefore, the precision will quantify the proportion of images that actually presented it over the total images that have been classified as such, including the erroneous ones. Again, the higher the value, the better. In the best case, FP=0 and therefore the precision will be the unit.

$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$

Recall. It quantifies the number of correct predictions of a certain category out of the total of those that really belong to that same category. In this case, it measures the images that the model has correctly classified with metastasis over the total images that actually present it. Recall is also known as true positive rate or sensitivity.

$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

F1-Score. It is a weighted average between precision and recall, so it takes into account both false positives (FP) and false negatives (FN). It is a less intuitive measure but more used than accuracy when evaluating and comparing various models.

$$F1 - Score = 2\frac{Recall * Precision}{Precision + Recall} \tag{6.4}$$

ROC curve. The ROC curve is one of the most relevant metrics for machine learning. It confronts the true positive rate (y axis) with the false positive rate (x axis), the latter being understood as the complementary probability of the **specificity**.

$$Specificity = \frac{TN}{TN + FP} = 1 - FPR \tag{6.5}$$

Where FPR is the false positive rate. The ROC curve is an excellent tool for observing the ability of the model to separate between classes. To deepen about this measure, the interested reader is referred to [71].

Area Under Curve (AUC). After studying the ROC curve, AUC is defined as the area under it. Therefore, it is possible to quantify with a metric the power of the model to separate between classes. As expected, the higher the AUC, the better the model.

6.2 Description of the proposed machine learning models

Throughout the different sections, a brief explanation of what feature extractors and classifiers are, as well as some examples of them with application in Computer Vision (Sections 5.1 and 5.2) has been given; training, validation and test procedures have been described, including cross-validation as a technique to lessen the effects of overfitting, and hyperparameter tuning (Sections 5.3.1 and 5.4); and the characteristics of the images to be classified have been illustrated (Section 6.1.2). In this section, the proposed model is described in detail. Actually, it is not exclusively a model, but a system of models that will be compared based on various metrics (see Section 6.1.3) in order to define which is the best. In short, the proposed system will serve as a benchmark to be able to conclude which type of model is more convenient for this particular problem.

The complete model is illustrated in Fig. 6.5. Each of its parts is detailed below, highlighting the relevant parameters and hyperparameters in each, as well as other relevant design decisions.

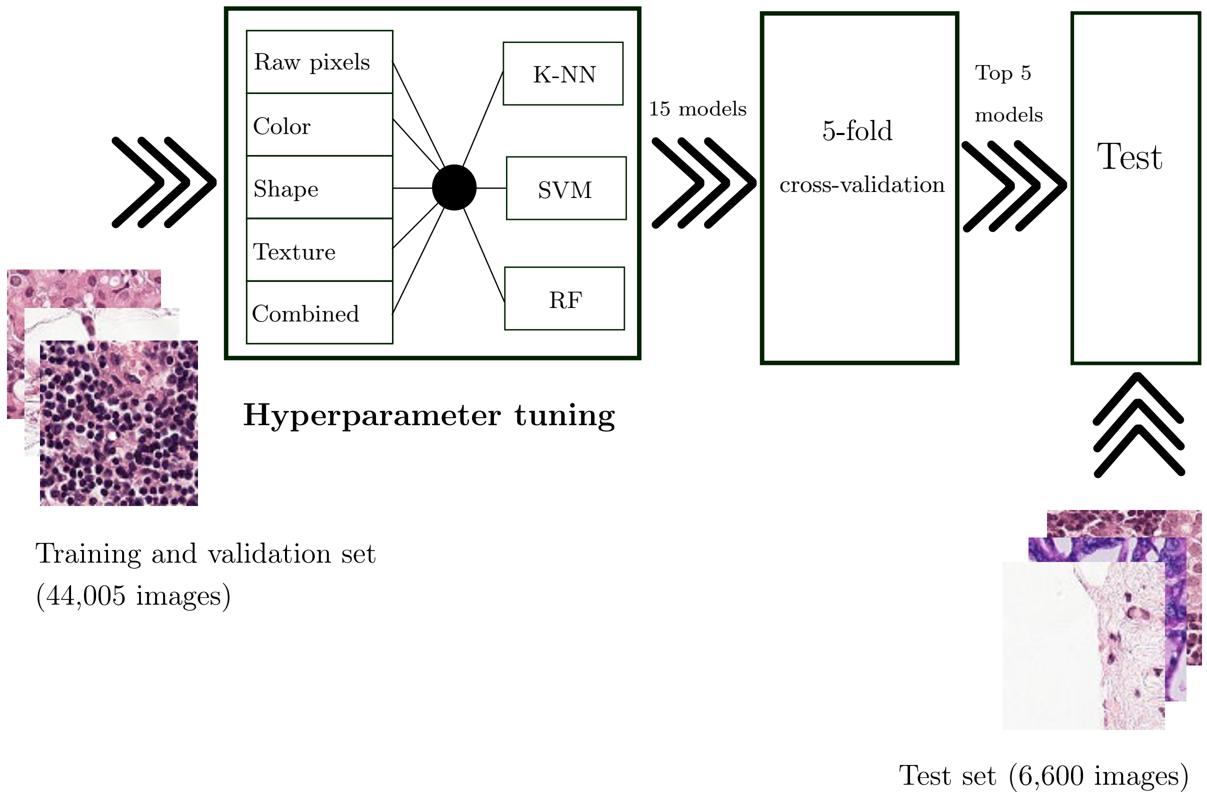


Figure 6.5: Complete system. The images of the training set are introduced to a subsystem consisting of five feature extractors and three classifiers, the parameters of which have been optimized by hyperparameter tuning. The fifteen possible models are validated by 5-fold cross-validation. The top five models are evaluated in a separate test set.

6.2.1 Features extractors

In Chapter 5 it was explained that machine learning models – and especially those linked to Computer Vision tasks – consisted of a feature extractor and a classifier. The presented model will consist of five feature extractors and three classifiers, combining all the possibilities until obtaining fifteen different models. Fig. 6.6 shows the structure of this specific part of the system.

More specifically, in Section 5.1 three types of feature extractors (color, shape and texture) were exemplified in four specific techniques: color histograms, Hu moments, Histogram of Oriented Gradients (HOG) and Haralick texture. In the proposed model, these different techniques are combined to obtain different feature extractors. In addition, with didactic intentions, one of the approaches will consist of not using any feature extractor, but directly introducing the images into the classifiers.

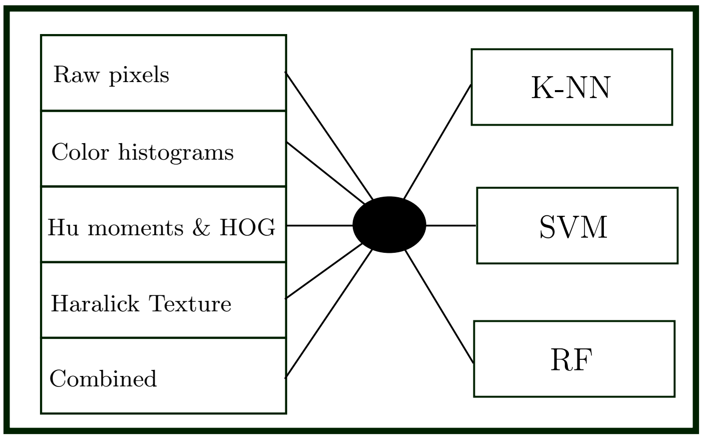


Figure 6.6: Explanatory diagram of the features extraction and classification subsystem. The “ Combined ” extractor integrates all of the above except Raw pixels (i.e. color histograms, Hu moments, HOG and Haralick Texture).

In total, therefore, five feature extractors are proposed:

Raw images (without feature extraction). The 96x96 pixel images are flattened, generating a feature vector of 9,216 values. Features are not extracted, so images are introduced directly into the classifiers. The memory occupation of each image will therefore be 27 KBytes (considering that each pixel is described by three colors (RGB) and each color is encoded with 8 bits).

Color feature extractor. Color from images is extracted using color histograms. The HSV color space is used, so a transformation from RGB to HSV space is performed. The number of “bins” chosen for each of the three histograms (Hue, Saturation, Value) is 16. Each image is described on average with a 16 KBytes feature vector.

Shape feature extractor. The information about the shape is obtained by concatenating the Hu moments and the HOGs of each image. Regarding the Hu moments, as mentioned in Section 5.1.2, before calculating them it is necessary to convert the images to grayscale. Regarding the HOGs, the parameters recommended in the original paper are maintained, using 8 "bins" and the L2-Hys normalization. In this particular problem, each image has been divided into 8x8 pixel blocks, resulting in 12 blocks. In each of these 12 blocks, one HOG is computed for each 2x2 pixel cell, resulting in 6 HOGs for each block. These two choices are justified by obtaining empirically reasonable results. Each image is described on average with a 72.05 KBytes feature vector that, as observed, occupies more memory than the raw pixels.

Texture feature extractor. The Haralick texture of each image is obtained. Again, for the calculation of the Haralick texture it is necessary that the image is in grayscale, so the appropriate transformation is performed. Moreover, instead of keeping the GLCM matrix, the mean of the columns is calculated to reduce dimensionality and memory occupation. The Haralick texture does not require defining the value of any other parameter. Each image is described on average with a 0.1 KBytes feature vector, considerably less than in the rest of the extractors

Color, shape and texture feature extractor. Color, shape, and texture extractors are concatenated, that is, color histograms, Hu moments, HOGs, and Haralick texture. The previously defined parameters are maintained. Each image is described on average with an 88.15 KBytes feature vector, which equals the sum of the previous extractors, excluding the raw images.

This is the feature extractor system, which will be combined with the classifiers. In the next chapter, the hyperparameters of each one are specified, optimized by means of hyperparameter tuning.

6.2.2 Classifiers

The extracted features will now be introduced into a classifier, which will determine which category each feature vector belongs to and, thus, each image.

However, prior to the description of the classifiers, it is relevant to highlight the **pre-processing** carried out on the feature vectors and the labels. Pre-processing is a common requirement in any machine learning model. In the present work the framework used in the programming of the models is scikit-learn, in which the classifiers are very sensitive to the distributions that the input data follow. In general, classifiers will not behave correctly if the introduced feature vectors do not follow – or at least resemble – the standard normal distribution, that is, a Gaussian with zero mean and unit variance. The idea is as follows: if any of the feature vectors has a considerably higher variance than the rest, it will dominate the classification and make it impossible for the classifier to learn correctly from other vectors. The transformation performed is known as standardization, and is applied to both training and test data.

In addition, the labels also undergo a transformation: one-hot-encoding. This consists of a “binarization” of the labels to encode them in a format familiar to the classifier. Above all, it is used when the labels are categorical – that is, they are text labels – and you want them to take numerical values in binary code. In the present model, the possible labels are "0" (No cancer) and "1" (Cancer). Therefore, the binarized labels will be 01 and 10, respectively. More information about one-hot-encoding and its application in machine learning can be found in [23].

Once the feature vectors and their labels have been obtained and pre-processed, the design of the classifiers used and the justification of the selected hyperparameters are described below.

After reviewing the state-of-the-art in Chapter 3, it is concluded that the chosen classifiers will be k-NN, SVM and Random Forest, as shown in Fig. 6.6. Section 5.2 summarizes how each one of them works, as well as its most relevant hyperparameters. In the previous section, up to five feature extractors were described, therefore, up to fifteen extractor-classifier combinations will be proposed in the present work.

Furthermore, hyperparameters should be selected for each of the classifiers in order to optimize the training and validation. To do this, hyperparameter tuning is used (see Section 5.4). Table 6.1 summarizes the hyperparameters that are to be optimized for each of the classifiers, as well as various value propositions for each one. The selected hyperparameter tuning method has been Random Search, using for each model six combinations of hyperparameters chosen at random.

Classifier	**Hyperparameters**	**Values**
K-Nearest Neighbors	Nº of neighbors (k)	3, 5, 7, 11, 21, 51
	Metric	Euclidean, Manhattan
	Algorithm	Ball-tree, KD-tree
Random Forest	Nº of trees	50, 100, 200
	Criterion	Gini, Cross-Entropy
Support Vector Machines	Kernel	Lineal, Polynomic (degree=2)

Table 6.1: Possible hyperparameter values for each of the fifteen extractor + classifier combinations.

In short, for each of the fifteen models there will be an optimized combination of hyperparameters from among those shown in Table 6.1. However, there are some hyperparameters that for design reasons will remain the same for all classifiers. These are:

- In k-NN the neighbors weight will always follow a distance-based rule.
- There is no upper or lower limit on tree depth required in Random Forest.
- There is also no limit on the number of features for Random Forest, as long as the number of features per tree is less than the total.

Finally, the combinations that best behaviour showed in the hyperparameter tuning are summarized in Table 6.2. Note that each combination of feature extractor and classifier constitutes a different model and the fine-tuning has been carried out for all possible fifteen models.

Feature Extractor	**Classifier**	**Hyperparameter**	**Values**
Raw pixels	k-NN	N º of neighbors (k)	7
		Metric	Manhattan
		Algorithm	Ball-tree
		Weights	Distance
	SVM	Kernel	Lineal
	Random Forest	N º of trees	200
		Criterion	Cross-Entropy
		Max depth	No
		Max features	No
Color	k-NN	N º of neighbors (k)	5
		Metric	Euclidean
		Algorithm	KD-tree
		Weights	Distance
	SVM	Kernel	Lineal
	Random Forest	N º of trees	200
		Criterion	Cross-Entropy
		Max depth	No
		Max features	No
Shape	k-NN	N º of neighbors (k)	5
		Metric	Manhattan
		Algorithm	KD-Tree
		Weights	Distance
	SVM	Kernel	Lineal
	Random Forest	N º of trees	200
		Criterion	Gini
		Max depth	No
		Max features	No
Texture	k-NN	N º of neighbors (k)	31
		Metric	Manhattan
		Algorithm	Ball-tree
		Weights	Distance
	SVM	Kernel	Lineal
	Random Forest	N º of trees	200
		Criterion	Cross-Entropy
		Max depth	No
		Max features	No
Color/Shape/Texture	k-NN	N º of neighbors (k)	11
		Metric	Manhattan
		Algorithm	Ball-tree
		Weights	Distance
	SVM	Kernel	Lineal
	Random Forest	N º of trees	200
		Criterion	Gini
		Max depth	No
		Max features	No

Table 6.2: Selected hyperparameter values for each of the fifteen extractor + classifier combinations.

It is important to note that in all the models that use SVM as a classifier, since the optimal kernel is linear, it was decided to simplify the SVM using the version C-Support Vector Classification (SVC). The reason is that SVM in its original version presents a quadratic relationship between the number of samples and the computational cost. Therefore, when working with this database, which is considerably large, using directly SVM meant that, for example, the hyperparameter tuning needed tens of hours to execute. Since computational cost is also a variable to consider, it was decided to simplify the system and use SVC, where not all possible distances are computed, but rather the most relevant ones. The documentation about SVC can be found in [16].

All models of the system are now defined. The following section describes the conditions under which each of the models is trained, validated and tested, as well as the criteria used to define which one is the best for the specific problem addressed in the present document.

6.2.3 Training, validation and test

Section 6.1.2 details the nature of the dataset used in this problem. In short, it consists of 50,605 images that contain sections of H&E-stained axillary lymph nodes.

The first partition is between the training and validation set and the test set. In 6.1.2 the partition of 44,005 images for training and validation and 6,600 for testing was justified, finding the split between the recommended values of 10% and 15%.

The 44,005 images with their respective labels are inserted into each of the fifteen models, made up of a feature extractor and a classifier. It is recalled that the classifiers have been optimized by hyperparameter tuning and that between the feature extractor and the classifier there is a pre-processing of both the feature vectors and the labels.

In this first phase, the fifteen models compete for a better cross-validation of the set of 44,005 images. The value $k = 5$ is chosen in the cross-validation, so that in each of the fifteen models the dataset of 44,005 images is separated into 5 folds of 8,801 images each. Next, five validations are performed, obtaining an accuracy score for each one. In each validation, one of the folds is used to evaluate and the other four are trained. In this way, five accuracy scores are obtained, from which the average is calculated. Section 5.3.1 explains generically what k-Fold cross-validation consists of. This procedure is repeated with the fifteen models, obtaining fifteen different accuracy scores.

For simplicity, the discriminating criterion for choosing which model is best will be the score obtained in the cross-validation. Thus, from the fifteen initial models, the five best will be chosen to be evaluated with the test set.

The test set consists of 6,600 images. In the previous phase, the five combinations of extractor + classifier that had the best results in cross-validation were selected. Here we will study the degree of success of each of these five models when generalizing them with another independent dataset (test set). This will be quantified with the metrics and indicators described in Section 6.1.3: confusion matrix, accuracy, precision, recall, F1-Score, ROC Curve, and AUC.

Chapter 7

Results

Once the system has been designed and the inputs characterized, this chapter presents the results obtained in the proposed models, both in the cross-validation and in the generalization to the test set.

It is important not to forget what the main objective of the system is, since it will serve to interpret the results obtained, especially when generalizing to the test set. It is recalled, therefore, that the proposed system aims to be able to detect metastatic tissue in patches of sections of axillary lymph nodes stained with H&E.

The present chapter will be divided into two parts: a first one referring to the validation results – where the fifteen initial models compete – and a second one referring to the test results – where the five best models of the previous validation compete.

7.1 Cross-validation results

In the first part of the system, the initial fifteen models obtain an accuracy score by means of a cross-validation with $k = 5$. This means that the set of 44,005 images is divided into 5 folds of 8,801 each, making a total of 5 validations from which the average is calculated. For simplicity, the only criterion used to select the best models is the accuracy value since, despite not giving excessive information, it does serve to discriminate, in general terms, which models classify best. Specifically, five of the fifteen models are selected for subsequent evaluation in the test set. Table 7.1 summarizes the accuracy scores of each of the fifteen models.

Feature extractor	Classifier	Accuracy (%)
Raw pixels	k-NN	66.96
	SVM	57.78
	Random Forest	67.90
Color	k-NN	89.43
	SVM	86.01
	Random Forest	89.31
Shape	k-NN	73.98
	SVM	53.01
	Random Forest	77.22
Texture	k-NN	80.52
	SVM	57.74
	Random Forest	81.62
Color/Shape/Texture	k-NN	77.11
	SVM	77.06
	Random Forest	84.05

Table 7.1: Accuracy score of the fifteen models in the cross-validation.

In conclusion, the five models that present the best results, ordered from higher to lower accuracy are:

- Color feature extractor + k-Nearest Neighbors (89.43%)
- Color feature extractor + Random Forest (89.31%)
- Color feature extractor + Support Vector Machines (86.01%)
- Color, shape and texture feature extractor + Random Forest (84.05%)
- Texture feature extractor + Random Forest (81.62%)

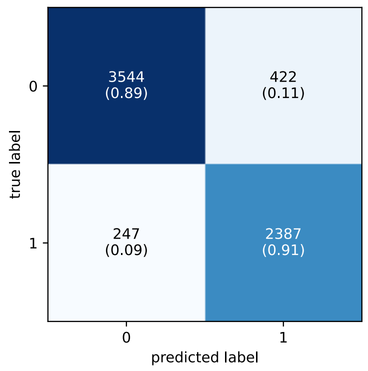
The meaning of these results will be discussed in detail in Chapter 8.

7.2 Test results

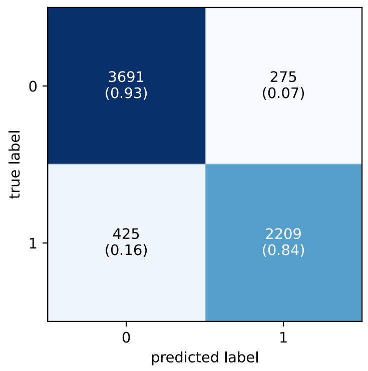
The top five models are now evaluated with the independent test set. This is made up of 6,600 images. To do so, first the selected models are retrained, this time using the 44,005 set of images exclusively for training, as no validation is required.

For the evaluation on the test set a cross-validation is not performed, unlike the previous phase. Instead, the images are classified using a single fold with all the 6,600 images, subsequently comparing these predictions with the true labels. For a more comprehensive comparison, all the metrics studied in Section 6.1.3 are used. For each of the five models, their confusion matrices are first presented, where the different types of solutions are observed for each extractor + classifier combination. Next, the value of some of the most relevant metrics in the study of machine learning models is indicated, such as precision, recall or AUC, in addition to displaying the ROC curves, on which the calculation of AUC is based. Finally, it is interesting to make a comparison of the models taking into account their computational cost, that is, the time invested in the extraction of features, training and evaluation of each image. This additional study is conducted at the end of this section.

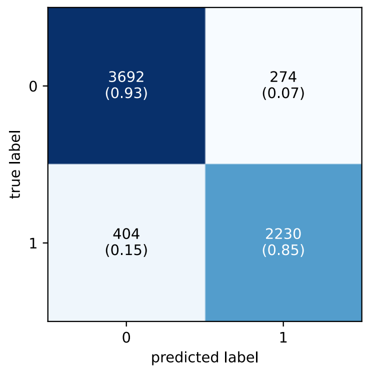
First, the confusion matrices generated in the evaluation of each of the five models on the test set are displayed in Fig. 7.1.



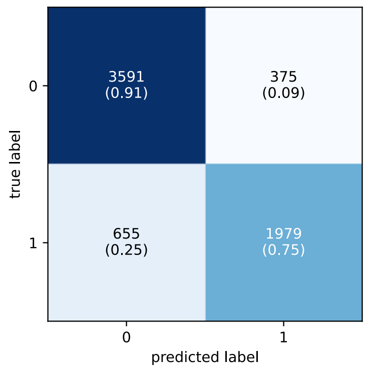
(a)



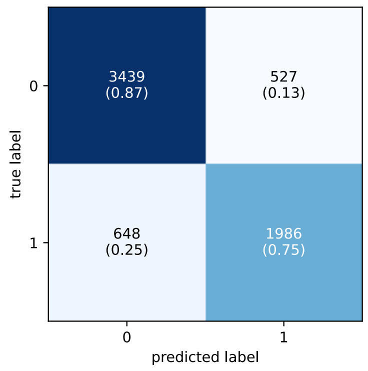
(b)



(c)



(d)



(e)

Figure 7.1: Confusion matrices for the models (a) Color + k-NN (b) Color + Random Forest (c) Color + SVM (d) Color/Shape/Texture + Random Forest (e) Texture + Random Forest.

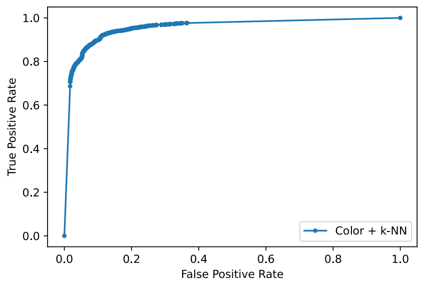
The x-axis represents the predictions and the y-axis the true labels. In each possible solution both the absolute and relative values are indicated. Fig. 6.4 in Section 6.1.3 showed the equivalence of this particular problem to the generic confusion matrix: true positive, true negative, false positive, false negative.

Moreover, Table 7.2 below summarizes the scores for each of the five models on the metrics defined in Section 6.1.3. Since some are rather complex, in the subsequent discussion in Section 8 they will be used mainly in comparative terms, that is, evaluating which of the models is better in a certain aspect based on their score in the metrics.

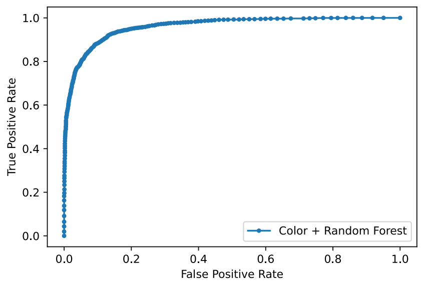
Model	Accuracy	Precision	Recall	F1-Score	AUC
Color + k-NN	**89.86**	84.97	**90.62**	**87.70**	95.5
Color + SVM	89.73	**89.06**	84.66	86.82	95.9
Color + Random Forest	89.29	88.68	83.86	86.29	**96**
Color/Shape/Texture + Random Forest	84.32	84.09	74.86	79.21	91.6
Texture + Random Forest	82.02	78.54	75.59	77.04	89.5

Table 7.2: Figures of merit for the compared models.

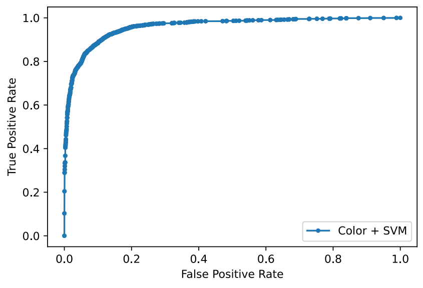
The ROC curves can be seen in Fig. 7.2. These face the true positive rate and the false positive rate. They are also used to calculate the AUC.



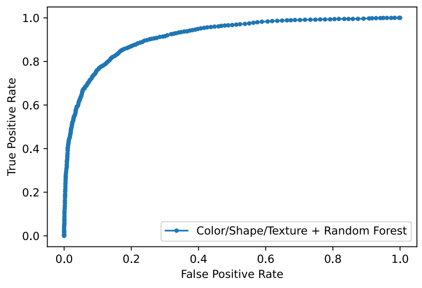
(a)



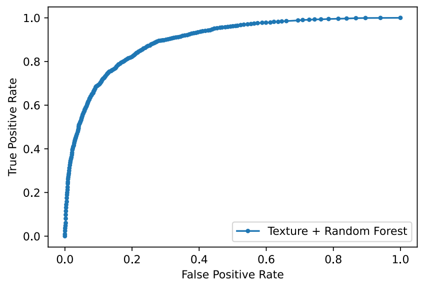
(b)



(c)



(d)



(e)

Figure 7.2: ROC curves for the models (a) Color + k-NN (b) Color + Random Forest (c) Color + SVM (d) Color/Shape/Texture + Random Forest (e) Texture + Random Forest.

Finally, as anticipated at the beginning of the section, the ability to classify is not the only criterion to take into account in a model. Computational cost imposes itself as another important consideration, especially in sophisticated and cutting-edge models where training, validations and test can take tens of hours or even days. Therefore, below, the performance of each of the five models is summarized in two graphs, facing their accuracy score – this metric is chosen for simplicity, since it is more intuitive – and the training and test times. Both feature extraction and fitting times, in the case of training, or classification times, in the case of the test, are included. Also, times are computed per image, assuming that the models take the same time on any image. Therefore, this time has been obtained by dividing the total time by the number of training or test images.

Table 7.3 summarizes these times. The values in *italics* are the ones used in the later graphs.

	Training			**Test**		
Model	**Feature extraction (s)**	**Fitting (s)**	**Total /Image (ms)**	**Feature extraction (s)**	**Classification (s)**	**Total /Image (ms)**
Color + k-NN	14.24	58.62	*1.65*	1.65	256.52	*5,87*
Color + SVM	14.24	20233.93	*460.13*	1.65	480.80	*10,96*
Color + RF	14.24	76.80	*2.07*	1.65	0.34	*0,04*
C/S/T + RF	3155.44	298.38	*78.48*	476.16	1.48	*10,85*
Texture + RF	1436.03	25.24	*33.20*	213.67	0.21	*4,86*

Table 7.3: Training and test times for the five best models obtained in the cross-validation. Training times are broken down into feature extraction and fitting, while test times into feature extraction and classification.

Once the training and test times per image have been defined in each of the models, they are visualized in two graphs

The first represents a comparison between the accuracy score in the test and the training time per image.

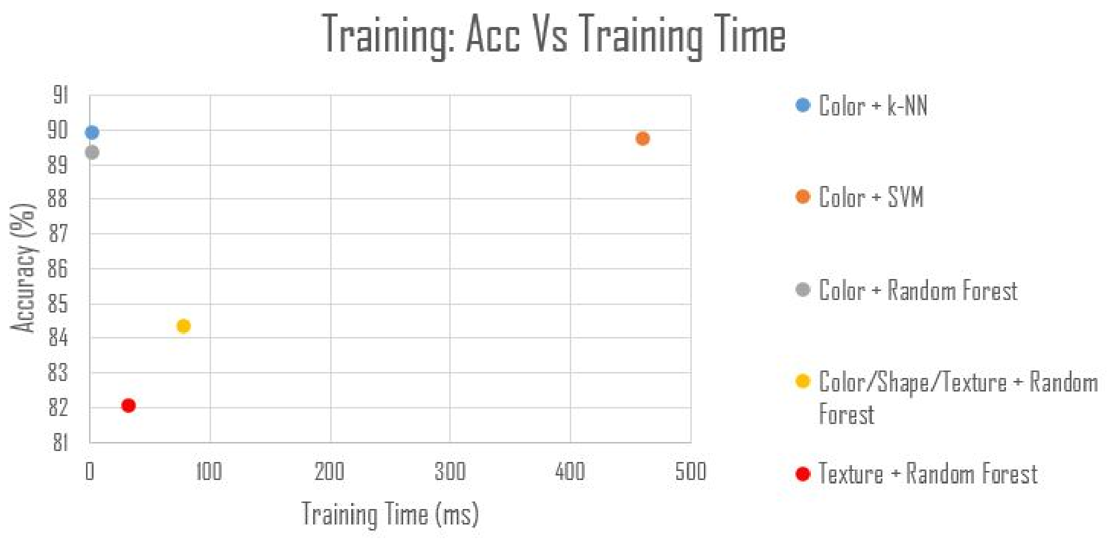


Figure 7.3: Comparison of the accuracy score and training time per image (in milliseconds) of the five best models.

Fig. 7.3, therefore, gives information on the training time per image necessary to obtain accuracy scores later in the independent test set.

The second keeps the same structure, but now comparing test times, instead of training times.

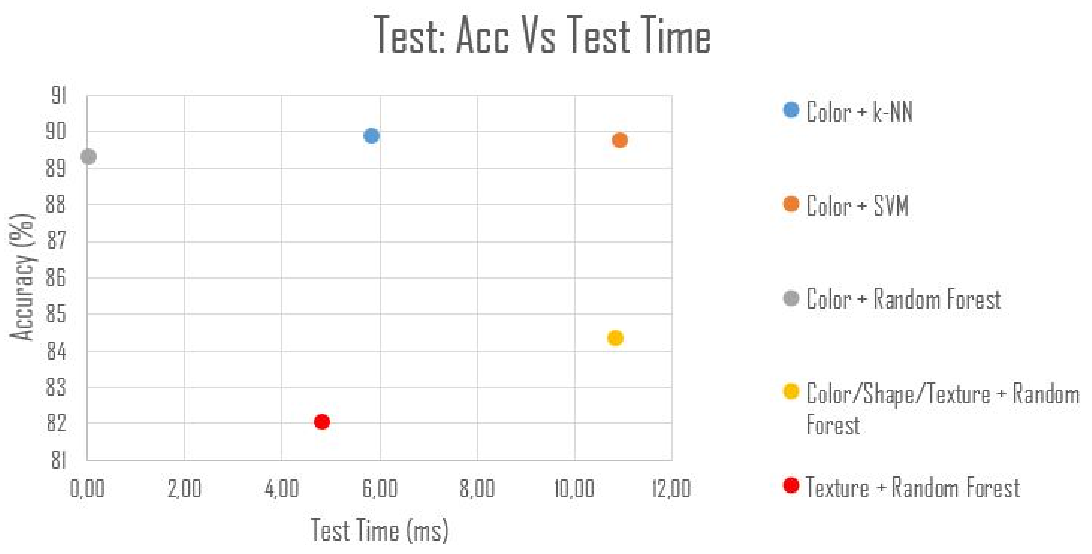


Figure 7.4: Comparison of the accuracy score and test time per image (in milliseconds) of the five best models.

Fig. 7.4 focuses on the evaluation times per image that the model invests to obtain a certain accuracy score on that same set of test data.

The results obtained in Fig. 7.3 and Fig. 7.4 will be discussed in depth in Chapter 8, along with the rest of the results. Specifically, for these two graphs, one of the analysis criteria will be the Pareto efficiency.

Chapter 8

Discussion

This chapter is divided into two parts: "General discussion" and "Comparison with other cutting-edge techniques". In the first, the results obtained in Chapter 7 are analyzed, that is, the result of the cross-validations and the test, emphasizing the meaning of the metrics and graphs that evaluate the ability of the models to classify the test set. On the other hand, the second section compares the performance of the models proposed in this work with another avant-garde technique: Convolutional Neural Networks (CNNs). Appendix A serves as an introduction to CNNs and gives an idea of its applicability in Computer Vision problems and, more specifically, in medical imaging.

8.1 General discussion

Throughout Chapter 7, an exhaustive comparison between the fifteen proposed models has been done. Now, the results are carefully analyzed.

8.1.1 Cross-validation results

The first part of Chapter 7 was devoted to the results obtained in the cross-validation, where the initial fifteen models competed. These results are shown in Table 7.1.

As expected, the alternative in which the feature extractor was not used (Raw pixels) is the one with the worst results in average terms. The reason is the one that has been mentioned recurrently throughout the document: however, as sophisticated the classifier may be, it will never be able to act exactly like the human eye. Classifiers require meaningful information and cannot work properly using only raw images. Regarding the different feature extractors, the one with the best behavior is the color feature extractor, based on color histograms. Furthermore, the texture feature extractor gets better accuracy scores on average than the shape feature extractor. In fact, the extractor that combines all the features seems to get good results just because it includes the color feature extractor. Therefore, the system shows that adding new features to the feature vector will not always improve the classification (i.e. adding texture and shape to the color extractor), but will improve it as long as the new features are representative and useful for identifying patterns. It can be concluded that the feature that best defines axillary lymph node patches stained with H&E is color. This makes sense and is precisely the reason why stains such as H&E are used: the difference between the structures in lymph nodes is evidenced by them, facilitating the differentiation between acidic and basic structures (see Section 4.3).

Regarding the classifiers, the one that shows the best results in average terms is Random Forest. In Section 5.2.3 it was studied that Random Forest is a very powerful classifier whose success was largely conditional on the features used. K-NN, on the other hand, presents similar results to Random Forest. In fact, the combination of color histograms with k-NN is the one with the best accuracy. Finally, SVM shows considerably worse results than the rest. This could be justified as a simplified version of SVM is used, known as SVC. Therefore, it appears that SVC performs reasonably well when the features used are meaningful (i.e. color) but it is unable to identify patterns and classify correctly when they are not as significant (i.e. shape and texture). This conclusion confirms that the detection of lymph node metastasis depends fundamentally on the color of the H&E-stained patches.

8.1.2 Test results

From the cross-validation, the five best models are selected. As a reminder, these are:

- Color feature extractor + k-Nearest Neighbors (89.43%)
- Color feature extractor + Random Forest (89.31%)
- Color feature extractor + Support Vector Machines (86.01%)
- Color, shape and texture feature extractor + Random Forest (84.05%)
- Texture feature extractor + Random Forest (81.62/%)

Section 7.2 described the performance of each of these models with the test, based on the metrics described in Section 6.1.3. In this section we aim to give meaning to these metrics, particularizing for our problem.

First, the **confusion matrices** give a general idea about the successes and errors of each model. The two errors that the model can make are: diagnosing with cancer a healthy lymph node (false positive) and not detecting cancer in a lymph node that does present metastases (false negative). Both errors are assumed to be equally serious, and therefore one model will not be better than the other based on which of the two errors it makes the least. Of the matrices in Figure 7.1, the model that best detects metastatic tissue is the one made up of the color feature extractor and SVM, with 93%. On the other hand, the one that best detects healthy tissue is the one formed by the color feature extractor and k-NN with 91%. This model, in fact, is the most balanced between the two types of errors and, therefore, the apparently most useful at an industrial level. The combination of color feature extractor and Random Forest is powerful at detecting cancer (93%), however, not so much at detecting healthy tissue (84%). The last two models are considerably worse and would therefore be discarded in an industrial application.

Table 7.2 summarizes some of the most significant metrics of machine learning applied to this problem. It is relevant to highlight that four of the five models have behaved similarly with the test set, comparing them with the values obtained in the cross validation. However, in the case of the model formed by the color extractor and SVM, when generalizing it to the test set the results are much better. Based on the metrics, it can be concluded that:

- The model that has the best **accuracy** is the one formed by the color extractor and k-NN. Therefore, in general terms, it will be the one that best classifies the lymph node sections. However, it does not give information on whether it is better to detect cancer or healthy tissue.
- The model with the best **precision** is the one formed by the color extractor and SVM. Therefore, it is the model that makes the least mistake when it comes to detecting cancer, since precision quantifies the ratio between the correct predictions of cancer and the total ones made with this verdict.
- The model with the best **recall** is the one formed by the color extractor and k-NN. Therefore, it is the model that has the best ability to detect tissues with metastases. This metric only takes into account patches with lymph nodes that present cancer. This means that, if the majority of patches potentially present cancer, this model would be the most convenient (e.g. a revision of patches of lymph nodes already classified by the pathologist as metastatic).
- If **F1-Score** is used – which serves as an average between precision and recall – the best model will be, again, the one formed by the color extractor and k-NN.
- However, if we look at the **AUC** metric, the best model is the one formed by the color extractor and Random Forest. AUC is based on the ROC curve and, despite being an unintuitive metric, it is often used to compare machine learning models.

It seems that it is not possible to accurately determine which of the three models is better. Its usefulness depends on the application (i.e. the k-NN model cancer more precisely when the majority of patches present cancer, but the SVM model is makes less mistakes on average) and, in any case, should be reviewed by an expert pathologist.

Finally, the trade-off between accuracy and **computational cost** is now evaluated. If the training time per image is taken into account (Fig. 7.3), the model formed by the color extractor and k-NN is the one that presents the best performance due to both high accuracy (89.86%) and low computational cost (1.65 ms per image). The reason is that the k-NN classifier is an instance model (see Section 5.2.1) and, therefore, there is no training as such, but the images are placed according to their feature vectors without there being a "learning" process. The combination of color extractor with Random Forest also performs considerably well (89.29%, 2.07 ms per image). The SVM color extractor model, however, shows a very high computational cost (460.13 ms per image). This is because SVM training time has a quadratic relationship with the number of features, which is very high in a Computer Vision problems. This problem was attempted to be solved using the simplified version of SVM known as SVC, however, as noted, it still shows very long CPU times.

Fig. 7.4, on the other hand, compared the accuracy of the models with the times dedicated to the evaluation of the test set. The ones that show better times are the models that use Random Forest, where the combination with color extractor stands out due to its high accuracy (89.29%) and low test time (0.04 ms per image). The k-NN color extractor model has reasonable times (5.87 ms per image). The SVM model has a somewhat higher computational cost, although the difference is not as high as in training (10.96 ms per image). In addition, in general, the more memory occupation the feature vectors that represent the images have, the longer their processing time will be, both in training and in tests.

Additionally, an analysis of the Pareto efficiency can be done in both figures. In Fig. 7.3 where the training times are compared, the Color + k-NN combination is identified as Pareto optimal, since it is the best in both computational cost and accuracy. The rest of the model would therefore be dominated by, at least, another model. Fig. 7.4, on the other hand, does not present Pareto optimal. The Color + k-NN combination dominates the rest in accuracy, while Color + Random Forest dominates in computational time.

Nothing is concluded about the two models made up of extractors of features other than color, as they perform considerably worse than the rest and, in any case, would be discarded for any industrial application.

8.2 Comparison with other cutting-edge techniques

One of the objectives indicated in Chapter 2 was to compare models based on traditional machine learning techniques (e.g. k-NN, SVM and Random Forest) with other cutting-edge techniques, specifically with Convolutional Neural Networks (CNNs). CNNs have become the method to beat in Computer Vision since they aspire to classify any type of problem with excellent results in terms of accuracy, however, at the cost of high computational times and the need to learn through huge databases. Appendix A introduces CNNs and defines the terms used in this section.

In the Motivation (Section 1.1), it was explained that, in parallel with the present work, the thesis linked to the final degree project in Telecommunications Technologies and Services Engineering was developed, whose author is also Javier Abad Martínez [1]. This thesis addresses the same problem as the current document, but proposing solutions based on CNNs. Therefore, this section compares the results obtained using both techniques: traditional machine learning and Convolutional Neural Networks.

It is important to highlight that this section does not aspire to be a frontal comparison, since the experimentation conditions were not the same and, therefore, it would not be fair. For example, for CNN five times more images were needed for training, and the same test set was not used, which makes any comparison of accuracy score unfair. In addition, the framework used in CNN training (TensorFlow) supports GPU-acceleration and, therefore, the processing speeds are multiplied by a factor of up to x50 when compared to CPU-based processing, such is the case of the present work.

Table 8.1 summarizes the training times, test times and accuracy scores of three model proposals based on CNNs, together with the results obtained by the three best models explained in this document. All three are architectures inherited from other developers (e.g. Google, Microsoft) to which fine-tuning is applied (see Appendix A) to adapt it to our problem. Next to the name of each network there is a reference to the original paper.

They can also be located in Fig. 7.3 and Fig. 7.4 in Section 7.2. The result is the one shown in Fig. 8.1 and 8.2.

Model	Training time per image (ms)	Test time per image (ms)	Accuracy (%)
VGG19 [24]	119.52	0.73	95.75
Inceptionv3 [53]	120.32	0.64	93.29
ResNet50 [59]	147.06	0.82	92.17
Color + k-NN	1.65	5.87	89.86
Color + SVM	460.13	10.96	89.73
Color + Random Forest	2.07	0.04	89.29

Table 8.1: Training time, test time and accuracy for the compared models.

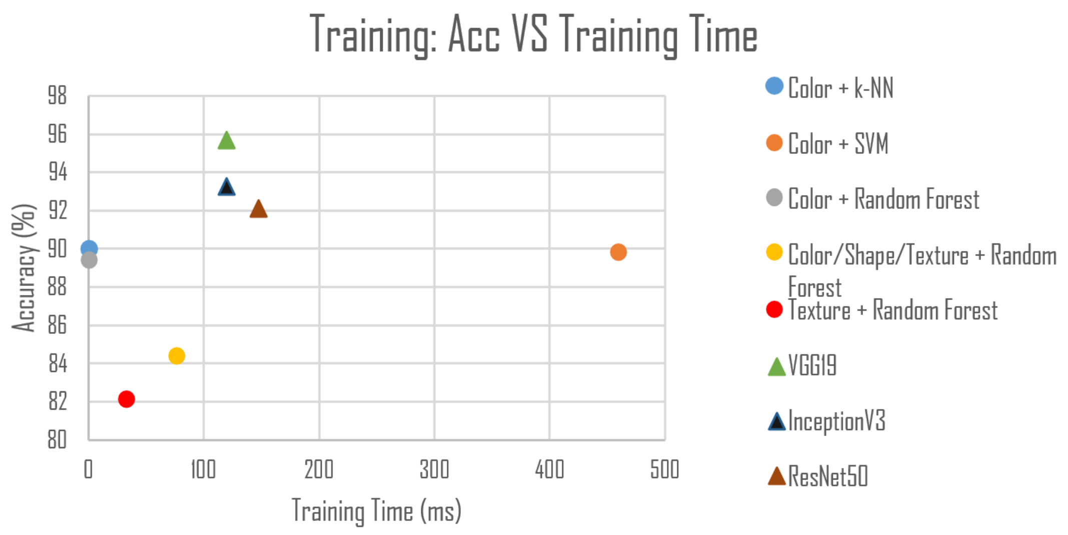


Figure 8.1: Comparison between traditional models and CNNs based on their accuracy and training times (in milliseconds). Circles represent the former, while triangles represent the latter

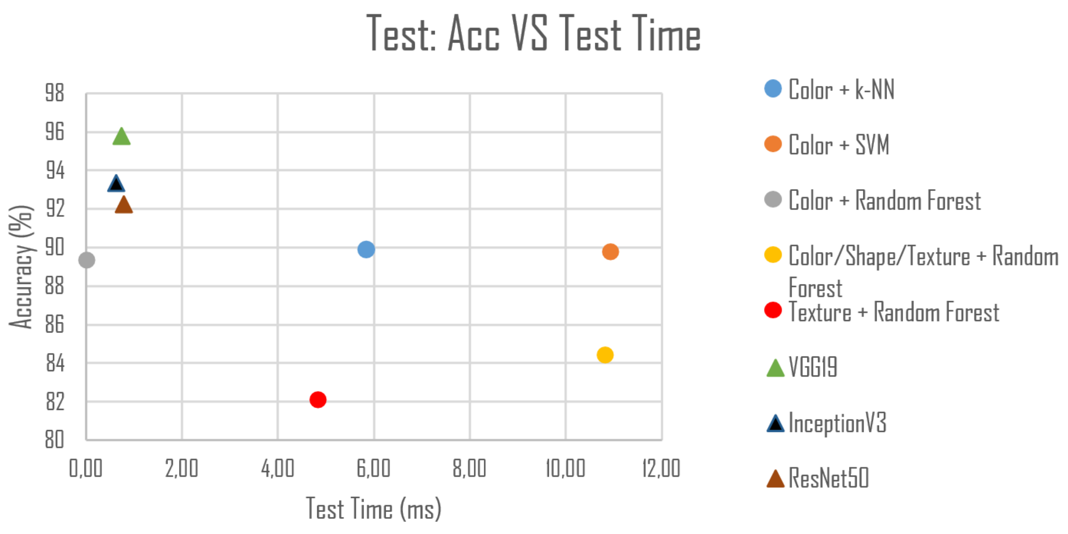


Figure 8.2: Comparison between traditional models and CNNs based on their accuracy and test times (in milliseconds). Circles represent the former, while triangles represent the latter

Models based on CNNs are indicated with triangles. All of them obtain considerably better accuracy scores than any of the previous models. Regarding training times, they are much higher than most models based on traditional machine learning, with the exception of those that use SVM as a classifier. This is due to the CNNs training process, based on "epochs", which implies that they adjust their models as many times as "epochs" have been indicated by the programmer. One of the problems of this system is the danger of overfitting, which is, in fact, one of the main criticisms of CNNs. Finally, the test times are reasonable, standing at better values than most models.

It is recalled that, in any case, CNNs have been accelerated by GPU. If they had not, the times could have been up to fifty times higher. This would imply that the training lasted several days. This is one of the reasons that CNNs have become so popular recently: accelerated GPU-based computing. However, in environments where there are time restrictions and there is no GPU, traditional machine learning models are most likely to be more useful, despite having lower accuracy scores. Similarly, special implementations of traditional machine learning models could also make use of GPUs, reducing times enormously. Furthermore, it is recalled that CNNs require large databases. Therefore, in case this resource is scarce, there can be interesting alternatives based, for example, on k-Nearest Neighbors, Support Vector Machines or Random Forest.

It can be concluded that the choice of model will depend on the environment and the application. If GPU acceleration is available, CNNs are most likely the best solution to a Computer Vision problem, however, overfitting must be controlled and tests must be done in independent test sets to fully validate the models. Conversely, when there is a temporary restriction – or GPU acceleration is not possible – and the accuracy score is not extremely relevant – for example, if it is later to be reviewed by an expert – it is possible that traditional machine learning models are more convenient.

Chapter 9

Limitations

This chapter highlights the assumptions, possible biases and simplifications taken in the development of the project. These come both from the database itself and from the techniques used and, therefore, limit the conclusions and generalizations that have been reached.

Regarding the original database, it had several limitations, many of them related to the simulation of the pathologist's exercise as the classifier of the patches. The database has been enriched with images that do show metastatic tissue in order to have a significant number of images in each of the categories. However, this diagnosis is not the usual one in the routine work of a pathologist, where, in most cases, the sentinel axillary lymph nodes (SLN) do not present metastases. Therefore, the comparison of the proposed models with the clinical reality of the expert pathologists is not totally fair. In addition, in the usual exercise of their profession, pathologists have more factors when defining their diagnosis and are not based exclusively on SLN sections. This, in short, makes comparing the performance of algorithms and pathologists more unfair.

On the other hand, the models have been specifically trained to exclusively discriminate between metastatic and non-metastatic tissue. Therefore, they do not have the ability to diagnose other diseases that could be present in the lymph nodes, such as lymphomas, sarcomas, or infection. The detection of these diseases is relevant in the routine diagnosis of pathologists and has not been incorporated in the present work. Therefore, again, the conditions in which this work has been carried out do not fully emulate clinical reality.

Furthermore, the images used are a simplification of the original database of the Camelyon16 Challenge. The original database was made up of WSIs of higher resolution and, therefore, with more information. The current one, on the other hand, is made up of patches resulting from sampling the WSI, saved in lower resolution formats, such as .tif. Additionally, not all the images available were used, but approximately 23% of the total.

Finally, the present work has preferred to give a more didactic vision about the machine learning models specialized in medical imaging. Therefore, none of the proposed models aspires to have an accuracy score higher than the most sophisticated techniques, but only sought to serve as a comparison with them, being able to identify the advantages and disadvantages of applying each technique. Therefore, other techniques that would surely have obtained better results, such as those based on deep learning – like CNNs – or on more complex methods like XGBoost, have been left out of the scope of this work.

Chapter 10

Conclusions

Finally, this chapter presents the final conclusions of the work. These are directly identified with the objectives established in Chapter 2. The conclusions are as follows:

- The diagnosis of the stage of breast cancer based on lymphatic spread is essential when defining a prognosis and treatment. Here the detection of metastasis in sentinel axillary lymph nodes (SLN) acquires special relevance. Therefore, its correct and premature diagnosis is essential to combat one of the main causes of death in women.
- The design of a model capable of detecting metastatic tissue from sections of SLN stained with H&E has been successfully developed. Therefore, the applicability of machine learning and, more specifically, of Computer Vision in extremely sensitive fields such as medical imaging is evident. Some of the proposed models have sufficient accuracy scores for its integration and use in hospitals, for example, helping in the treatment of patients by giving a premature diagnosis, so that the pathologist starts from a base when defining the final diagnosis. Another implementation could be the use of these algorithms after the pathologist's diagnosis, so that it can be contrasted and reviewed in the event of issuing different verdicts.
- It can be concluded from experimentation that the feature that best describes the SLN sections stained in H&E is color. Therefore, it would be convenient to use extractors with this feature, for example, color histograms. It has been shown that in Computer Vision problems, the k-NN and Random Forest classifiers obtain reasonable results in both accuracy and computational cost, with SVM being dominated in terms of this second. In addition, the use of cross-validation, hyperparameter tuning and specialized metrics in machine learning allow for good performance and a comprehensive evaluation of artificial intelligence models.
- A comparison between CNNs and other traditional machine learning techniques has been developed. Although it has not been totally fair, it has been possible to identify some CNN requirements that other techniques do not need, such as huge databases and high computational costs, which require GPU-acceleration. In any case, it is concluded that it is

important to be familiar with both spheres, the most sophisticated and the most traditional, and that the applicability of each one will depend on the specific problem and the available resources.

- Furthermore, this project has a dimension and utility linked to sustainable development, that is, it has an environmental, social and economic component. The author supports the integration of machine learning techniques into any field, especially the medical field. It has been demonstrated that democratizing these technologies has a clear social value, since they help to deal with extremely sensitive issues such as breast cancer diagnosis, serving as support for the pathologist and in no case serving as a substitute for this profession. In addition, computational cost has been studied as a variable to take into account due to the carbon footprint. Using simpler models like k-NN and Random Forest instead of CNNs has been shown to help minimize this impact.

Chapter 11

Recommendations and future work

Finally, in this chapter some possible future work routes are indicated, to improve the system proposed in this document and potentially obtain better results.

In Chapter 10 it is concluded that color is the most significant feature to describe the images used in this work. Therefore, it would be reasonable to design models with more sophisticated color extractors than color histograms, for example, Dominant Color Descriptor (DC), Chromaticity, Color Co-occurrence Matrix (CCM) or Color Correlograms. The interested reader is recommended to consult [42], where these methods are evaluated along with many others, as well as their advantages and disadvantages in terms of applicability.

Another way to improve the models proposed here would be through more complex classifiers. In this thesis some traditional machine learning techniques have been exposed and contrasted with CNNs, however, there is a whole spectrum to explore. For example, XGBoost is another method based on decision trees that usually shows very positive results. It is recommended to consult the original XGBoost paper to go deeper [9].

Furthermore, it would be interesting to refine the proposed model, for example, by adding filters that highlight the colors of the lymph node sections and thus making it easier to identify the patterns. The idea is to improve the brightness and saturation of the images, as this could potentially have an impact on better classification capacity and, ultimately, higher accuracy scores.

Finally, some of the limitations raised in Chapter 9 could be overcomed. The current data base is a simplification of the original (Camelyon16 Challenge), therefore, to make sure of the classification capacity of the proposed models, they could be tested with these WSIs that, in addition, will have a better resolution and will give more information for the classification. Obviously, this change would involve incorporating a phase of image and database treatment into the system, since the handling of WSIs is more complex than that of images in .tif format.

In the case of implementing all of the above, it would result in a more real system and potentially with greater classification capacity. This would involve, in short, a more reliable application in hospitals.

Appendices

Appendix A

Introduction to Convolutional Neural Networks

A.1 How Neural Networks work

Neural networks fall within what is known as deep learning, a subdiscipline of machine learning that uses neural network-based models for regression and classification problems. This appendix does not aspire to exhaustively describe the functioning of neural networks, but merely comments in broad terms on how they classify, as well as their learning mechanism.

The classification and learning of neural networks is based on two complementary processes: forward-propagation and back-propagation. Forward-propagation is a "forward" process, that is, it occurs from the beginning to the end of the network. The network receives matrices with information about an object as its input – for example, the pixels of an image – and will perform a series of operations – depending on the characteristics of the particular network – to ultimately result in a value that will determine the category to which the object we are trying to classify belongs. On the other hand, back-propagation is a "backwards" process. As we have seen, the object has been classified thanks to a succession of operations carried out by each of the layers of the network. These operations are characterized by parameters – known as weights – whose value will be such that the final predictions will be correct as many times as possible. Back-propagation will therefore be the process by which these parameters are adjusted and take optimal values to generate the most accurate predictions. It is necessary to delve a little deeper into these two mechanisms, so each of them is described separately below.

As explained above, forward-propagation is the mechanism by which the neural network makes a decision based on the information it receives as input, that is, it is the pure classification process. It is relevant now to inquire about how the class to which the object belongs is "calculated". This is achieved thanks to a succession of layers. In its simplest form, each layer can be characterized by a weight matrix W that is multiplied by the input information matrix. Next, a non-linear function is applied to the result of the product – generally ReLU, softmax or sigmoid, depending on the objective and the network – which will result in a new matrix whose values will be more informative than before and, therefore, be easier to find patterns that allow classification. This non-linear function is known as activation. Fig. A.1 shows a diagram of how the layers of a neural network work in their simplest version. The input variables are expressed as x_i, the parameters of the layer being the weights w_i and its independent term b. The activation function is f. It is observed that the operation of each layer is summarized in a matrix product to which the activation function is applied to model possible non-linear relationships.

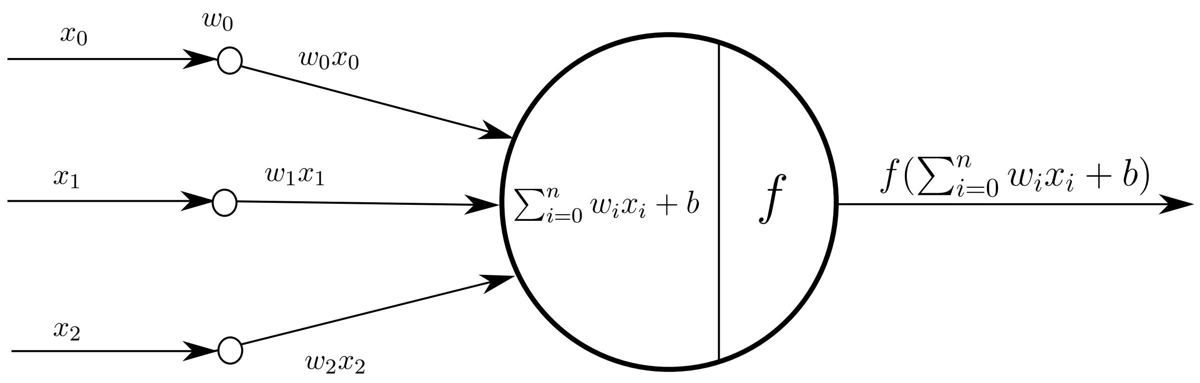


Figure A.1: Functioning and parameters of a neuron.

It is easy to see the analogy of the layers of a neural network with feature extractors seen throughout this document. In principle, the deeper the network, the more accurate the prediction. Finally, the last layer or the last layer block will serve as a classifier, so that its output is a value from 1 to N, where N is the number of classes. Fig. A.2 shows a simple implementation of this last layer. The structure is the same as the previous one, only in this case the output can only take two possible values, which will be the existing classes for this particular case. The output is, in short, the prediction of the model.

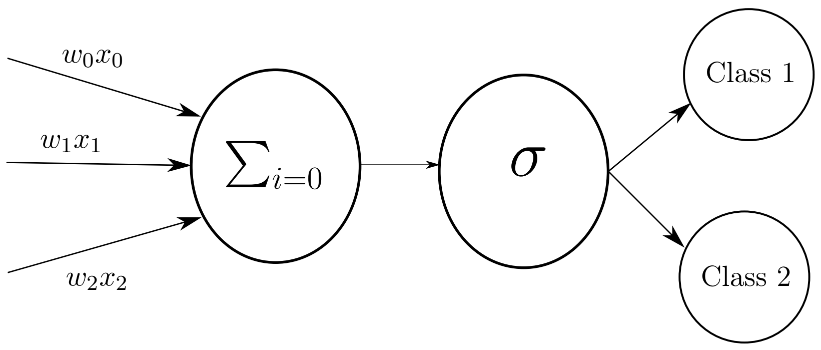


Figure A.2: Example of a classification layer. In this specific network, the activation function is sigmoid (σ) with two possible classes ($N = 2$).

It now remains to be defined how the weight values w_i are adjusted so that the predictions are as correct as possible. This is accomplished by the back-propagation mechanism. Neural networks are included within the area of supervised learning, therefore, in the training phase, the predictions are compared with the true labels of the objects. The differences between the predicted and the truth are quantified by means of a loss function that will have to be minimized.

Therefore, the weights w_i of the layers of a neural network will be those that minimize this loss function.

The process of minimization and, consequently, of optimization of the weights, is not trivial, and is based on what is known as optimization by gradient descent. The interested reader can consult the technical information for calculating the gradient descent in [49]. Furthermore, gradient descent is not the only method of parameter optimization. There are many alternatives and particular solutions existing in the literature, where SGR, Adam or RMSprop stand out, among others. Furthermore, the choice of the optimizer is not the only decision to be made by the programmer regarding the learning process. It is also necessary to choose the hyperparameters of the optimizer itself, in addition to the loss function, among other things.

This has been a considerably simplified explanation of neural networks, however, sufficient to understand their meaning in the problem posed in this document. The following section of this Appendix summarizes one type of neural network: Convolutional Neural Network (CNNs). Here you will see its application in Computer Vision and, consequently, in medical imaging.

A.2 The Convolutional Neural Network

In the previous section we have superficially explained what neural networks consist of, using the simplest layer, consisting of a matrix product and an activation function. However, in more complex problems it is necessary to build more sophisticated networks. This is the case of Computer Vision and, more specifically, of medical imaging. In this context, a specialized tool in the neural network is required: convolutional blocks.

Convolutional blocks are made up of a set of layers that manage to extract significant features from an image. Next, each part of a convolutional block is explained in a summarized way.

Convolutional layer. It is in charge of extracting the features of the image. It is based on filters called kernels. Kernels define the set of pixels on which the convolution operation is performed. In this way, a kernel goes through the image convolving the different sections of it with a filter, whose values are the parameters to be optimized. The movement of the kernel throughout the image is defined by the stride, which is the number of pixels that move the filter in the different dimensions travelling through the image. Fig. A.3 serves as an example of kernel operation in a section of the image:

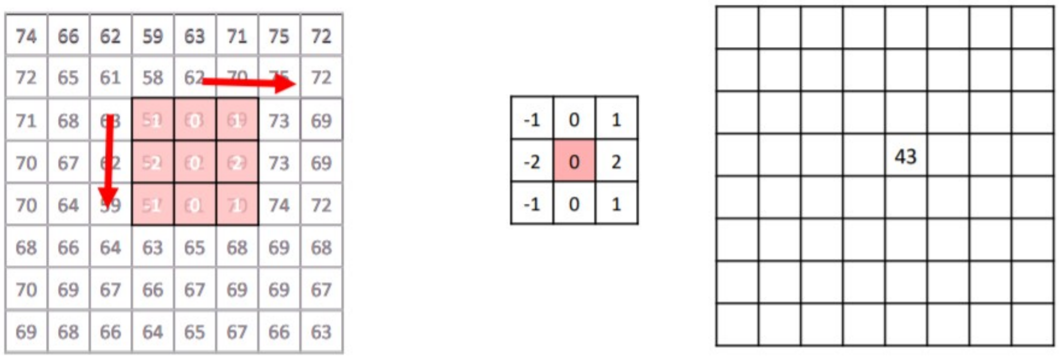


Figure A.3: Convolution of an image channel with a 3x3 size kernel. On the left, the image with the convolved section highlighted; in the center, the kernel weights to optimize; on the right, the result of the operation.

The image on the left is the input, the image in the middle is the kernel, and the image on the right is the output. For the value shown specifically, the 3x3 pixel region has been multiplied by the 3x3 kernel using one-to-one scalar products. The kernel application in Computer Vision is used to extract information related to the shapes, colors and textures of the image. Furthermore, as the number of convolutional layers increases – and therefore the network becomes deeper – the level of detail increases. In this sense, the first layers will be able to identify the most superficial contours in the image shapes, while the deepest blocks recognize more complex shapes and with a higher level of detail.

Padding. It may be the case that the introduced image has a number of pixels that is not a multiple of the kernel size. In this case, there are two solutions: zero-padding – in which a zero outline is added to the image – or valid-padding – based on dispensing with the part of the image where the filter does not fit. For example, if the filter is 2x2 in size and our 3x3 image (in grayscale), zero-padding would add zeros such that the image would be 4x4, while valid-padding would drop pixels, leaving the image at 2x2.

Activation. As in any neural network, CNNs use an activation to model non-linear relations. The typically used function is ReLU.

Pooling. In order to reduce the number of convolutions and, therefore, the computational cost, a pooling layer is added to the end of each block that reduces dimensionality. Pooling has the ability to leave the most relevant features, which are invariant to position and rotation. In this way, despite reducing the information, it will be possible to train the model correctly. There are two popular pooling alternatives: max pooling and average pooling. Max pooling takes the maximum value of the defined section, while average pooling computes the mean. Fig. A.4 serves as an example of both techniques.

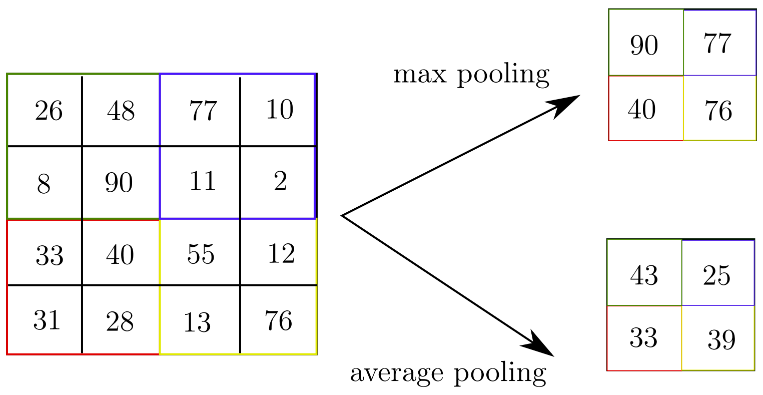


Figure A.4: Exemplification of the operations of max pooling and average pooling

Thus the concept of convolutional block has been roughly defined. The convolutional neural network will be made up of several convolutional blocks, whose parameters – that is, the values of the kernels – will be optimized in the learning process. The convolutional blocks therefore serve as feature extractors. To these must be added a top-level that will serve as a classifier. These are usually made up of simple layers – like the ones explained in the previous section – called dense or fully-connect layers. The final activation function – which is usually sigmoid or softmax – will be the one that defines which category the image belongs to. Fig. A.5 shows the complete structure of a CNN that classifies means of transport.

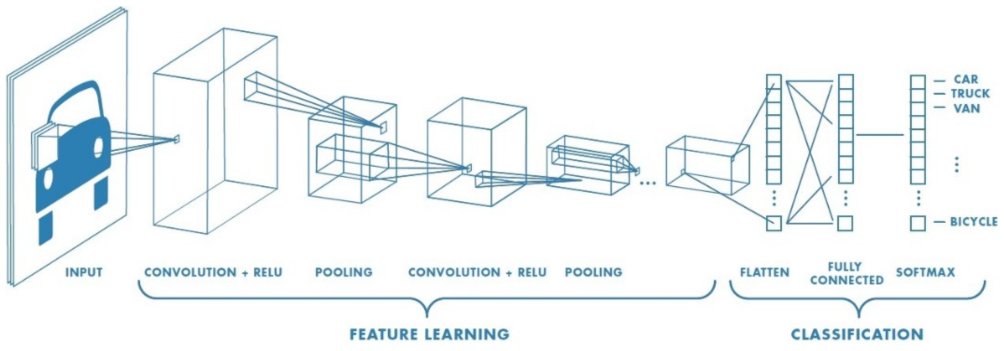


Figure A.5: Example of a Convolutional Neural Network. Two of the convolutional blocks are shown, consisting of a convolution layer, ReLU activation and pooling. A classifier consisting of a flatten, a fully-connect layer and softmax activation for the classification is added. The network classifies the input among various means of transport. Retrieved from *Basics of the Classic CNN*, by C. Churh Chatterjee ,2019 ,`https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add`. Copyright by Towards Data Science.

A.3 Specialized deep learning terminology

Finally, this section serves as a glossary for some specific terms of CNNs that are used throughout the thesis.

ReLU: rectified linear unit. Returns as output the maximum between the input and zero. Therefore, it is understood as a threshold function that discriminates negative values – giving them a value of zero – and allowing positives to pass without altering their value.

Softmax: activation function that converts a vector of real values into another of values belonging to the interval [0,1] whose sum is 1. Typically used in classification.

Epoch: In the process of learning of neural networks, in the first place, the set of input objects is divided into **batches**. The network will learn, that is, it will adjust the value of its weights as it classifies the images of the batches. When all the training images have already been classified it is said that an epoch has passed. In neural network optimization techniques, this process is repeated as many times as the programmer indicates, repeatedly classifying the same training images and adjusting the parameters accordingly.

Transfer learning: it consists of reusing the knowledge of neural network architectures used for particular problems in our problem. It makes it possible to customize these models and create considerably powerful networks, since they take advantage of networks whose parameters have already been optimized. Once the network is inherited, a different top-level classifier can be added depending on the categories of our problem. Some popular models applicable to virtually any problem are VGG19, ResNet, or Inceptionv3.

Fine tuning: transfer learning approach in which, in addition to adding a classifier to inherited architectures, the parameters of the deepest layers are trained. It is based on the idea that two different problems will be able to keep the same initial layers – since they are not highly detailed – but the deeper layers will necessarily have to be different, as they try to find concrete and specific shapes of the image. Therefore, it is very useful to inherit, for example, the ResNet structure and train only the last layers, in addition to adding a top-level.

Bibliography

[1] J. Abad. “Automatic detection of metastatic tissue in lymph node sections of breast cancer patients using Convolutional Neural Networks”. B.S. Thesis. Universitat Politècnica de València (UPV), 2020 (cit. on pp. 2, 4, 52).

[2] E. Alpaydin. *Introduction to Machine Learning.* 3rd ed. Cambridge, Massachusetts: The MIT Press, 2010 (cit. on p. 15).

[3] V. Antun, F. Renna, C. Poon, B. Adcock, and A. Hansen. “On instabilities of deep learning in image reconstruction and the potential costs of AI”. In: *Proceedings of the National Academy of Sciences* (May 2020). DOI: `10.1073/pnas.1907377117` (cit. on p. 10).

[4] J. Bergstra and Y. Bengio. “Random search for hyper-parameter optimization”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 281–305 (cit. on p. 29).

[5] L. Breiman. “Random Forests”. In: *Machine Learning* 45.1 (Oct. 2001), 5–32. ISSN: 0885-6125. DOI: `10.1023/A:1010933404324` (cit. on p. 25).

[6] Cancer.Net. *Breast Cancer Guide: Introduction.* Tech. rep. American Society of Clinical Oncology (ASCO), July 2019 (cit. on pp. 11, 12).

[7] G. C. Cawley and N. L. C. Talbot. “On over-fitting in model selection and subsequent selection bias in performance evaluation”. In: *The Journal of Machine Learning Research* 11 (2010), pp. 2079–2107 (cit. on p. 28).

[8] The Cancer Statistics Center. *How Common Is Breast Cancer?* Tech. rep. The American Cancer Society, Sept. 2019 (cit. on p. 11).

[9] T. Chen and C. Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: Aug. 2016, pp. 785–794. DOI: `10.1145/2939672.2939785` (cit. on p. 59).

[10] J. Shiny Christobel and C. N. S. Vinoth Kumar. “Feature based breast cancer detection using mammographic images”. In: 2017 (cit. on p. 9).

[11] M. Claesen and B. De Moor. “Hyperparameter search in machine learning”. 2015 (cit. on p. 29).

[12] T. M. Cover and P. E. Hart. “Nearest neighbor pattern classification”. In: *IEEE Transactions on Information Theory* 13 (1967), pp. 21–27 (cit. on p. 20).

[13] G. Currie, E. Hawk, E. Rohren, A. Vial, and R. Klein. “Machine learning and deep learning in medical imaging: Intelligent imaging”. English. In: *Journal of Medical Imaging and Radiation Sciences* 50.4 (Dec. 2019), pp. 477–487. ISSN: 1876-7982. DOI: 10.1016/j.jmir.2019.09.005 (cit. on p. 9).

[14] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1 (cit. on pp. 17, 18).

[15] C. Désir, S. Bernard, C. Petitjean, and L. Heutte. “A Random Forest Based Approach for One Class Classification in Medical Imaging”. In: *Machine Learning in Medical Imaging - Third International Workshop, MLMI 2012, Held in Conjunction with MICCAI 2012, Nice, France, October 1, 2012, Revised Selected Papers*. Ed. by F. Wang, D. Shen, P. Yan, and K. Suzuki. Vol. 7588. Lecture Notes in Computer Science. Springer, 2012, pp. 250–257. DOI: 10.1007/978-3-642-35428-1_31 (cit. on p. 9).

[16] scikit-learn developers. *C-Support Vector Classification*. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. [Online; accessed May-2020]. July 2019 (cit. on p. 42).

[17] P. Dollár, S. Belongie, and P. Perona. “The Fastest Pedestrian Detector in the West”. In: Sept. 2010, pp. 1–11. DOI: 10.5244/C.24.68 (cit. on p. 18).

[18] B. Ehteshami Bejnordi et al. “Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer”. In: *Journal of the American Medical Association* 318 (Dec. 2017), pp. 2199–2210. DOI: 10.1001/jama.2017.14585 (cit. on p. 32).

[19] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. “Object Detection with Discriminatively Trained Part-Based Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1627–1645 (cit. on p. 18).

[20] X. Glorot, A. Bordes, and Y. Bengio. “Deep Sparse Rectifier Neural Networks”. In: vol. 15. Jan. 2010 (cit. on p. 10).

[21] J. Griffin and D. Treanor. “Digital pathology in clinical use: Where are we now and what is holding us back?” In: *Histopathology* 70 (Jan. 2017), pp. 134–145. DOI: 10.1111/his.12993 (cit. on p. 13).

[22] R. M. Haralick, K. Shanmugam, and I. Dinstein. “Textural Features for Image Classification”. In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-3.6 (1973), pp. 610–621 (cit. on p. 19).

[23] S. Harris and D. Harris. *Digital design and computer architecture: arm edition.* Morgan Kaufmann, 2015, p. 129. ISBN: 978-0-12-394424-5 (cit. on p. 39).

[24] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778 (cit. on pp. 10, 53).

[25] M. K. Hu. “Visual pattern recognition by moment invariants”. In: *IRE Transactions on Information Theory* 8 (1962), pp. 179–187 (cit. on p. 17).

[26] R. Iglesis, R. Sandoval, R. Schwartz, and R. Olguin. “Hispatological analysis of the sentinel lymph node. Techniques and results”. In: *Revista Chilena de Cirugía* 54 (2002), pp. 380–3 (cit. on p. 12).

[27] J. Yang, D. Zhang, A. F. Frangi, and Jing-yu Yang. “Two-dimensional PCA: a new approach to appearance-based face representation and recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.1 (2004), pp. 131–137 (cit. on p. 15).

[28] A. K. Jain and A. Vailaya. “Image retrieval using color and shape”. In: *Pattern Recognition* 29 (1996), pp. 1233–1244 (cit. on pp. 16, 17).

[29] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning.* Vol. 112. Springer, 2013 (cit. on pp. 27, 28).

[30] N. Japkowicz. “Why Question Machine Learning Evaluation Methods ? (An illustrative review of the shortcomings of current methods)”. In: 2006 (cit. on p. 35).

[31] S. Jeong, C. Won, and R. Gray. “Image Retrieval Using Color Histograms Generated by Gauss Mixture Vector Quantization”. In: *Computer Vision and Image Understanding* 94 (Nov. 2003). DOI: `10.1016/j.cviu.2003.10.015` (cit. on p. 17).

[32] D. Krag et al. “Technical outcomes of sentinel-lymph-node resection and conventional axillary-lymph-node dissection in patients with clinically node-negative breast cancer: results from the NSABP B-32 randomised phase III trial”. In: *The lancet oncology* 8 (Nov. 2007), pp. 881–8. DOI: `10.1016/S1470-2045(07)70278-4` (cit. on p. 12).

[33] A. Krizhevsky, I. Sutskever, and G. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Neural Information Processing Systems* 25 (Jan. 2012). DOI: `10.1145/3065386` (cit. on p. 10).

[34] M. Kuhn and K. Johnson. *Applied predictive modeling.* Vol. 26. Springer, 2013 (cit. on p. 27).

[35] Y. Lecun, Y. Bengio, and G. Hinton. “Deep Learning”. In: *Nature* 521 (May 2015), pp. 436–44. DOI: 10.1038/nature14539 (cit. on p. 9).

[36] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on p. 1).

[37] R. A. Lerski et al. “Computer analysis of ultrasonic signals in diffuse liver disease.” In: *Ultrasound in medicine & biology* 5 4 (1979), pp. 341–50 (cit. on p. 19).

[38] “HSV Color Space”. In: *Encyclopedia of Microfluidics and Nanofluidics*. Ed. by D. Li. Boston, MA: Springer US, 2008, pp. 793–793. ISBN: 978-0-387-48998-8. DOI: 10.1007/978-0-387-48998-8_656 (cit. on p. 17).

[39] S. Z. Li, XiaoGuang Lu, Xinwen Hou, Xianhua Peng, and Qiansheng Cheng. “Learning multiview face subspaces and facial pose estimation using independent component analysis”. In: *IEEE Transactions on Image Processing* 14.6 (2005), pp. 705–712 (cit. on p. 15).

[40] J. O’Connor, C. Rose, J. Waterton, G. Parker, and A. Jackson. “Imaging Intratumor Heterogeneity: Role in Therapy Response, Resistance, and Clinical Outcome”. In: *Clinical cancer research : an official journal of the American Association for Cancer Research* 21 (Nov. 2014). DOI: 10.1158/1078-0432.CCR-14-0990 (cit. on p. 19).

[41] S. Paschalakis and P. Lee. “Pattern recognition in grey level images using moment based invariant features”. In: *Image Processing And Its Applications, 1999. Seventh International Conference on (Conf. Publ. No. 465).* Vol. 1. 1999, 245–249 vol.1 (cit. on p. 17).

[42] J. M. Patel and N. C. Gamit. “A review on feature extraction techniques in Content Based Image Retrieval”. In: *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET).* 2016, pp. 2259–2263 (cit. on p. 59).

[43] N. Pathan and M. E. Jadhav. “Medical Image Classification Based on Machine Learning Techniques”. In: *Advanced Informatics for Computing Research.* Ed. by A. K. Luhach, D. S. Jat, K. Bin G. Hawari, X. Gao, and P. Lingras. Singapore: Springer Singapore, 2019, pp. 91–101. ISBN: 978-981-15-0108-1 (cit. on p. 10).

[44] S. Pereira, A. Pinto, V. Alves, and C. A. Silva. “Brain Tumor Segmentation Using Convolutional Neural Networks in MRI Images”. In: *IEEE Transactions on Medical Imaging* 35.5 (2016), pp. 1240–1251 (cit. on p. 10).

[45] M. Prasad and V. Jarugumalli. “Performance Evaluation: Ball-Tree and KD-Tree in the Context of MST”. In: vol. 62. Oct. 2012. DOI: 10.1007/978-3-642-32573-1_38 (cit. on p. 22).

[46] Q. Chen, E. Petriu, and X. Yang. “A comparative study of Fourier descriptors and Hu’s seven moment invariants for image recognition”. In: *Canadian Conference on Electrical and*

Computer Engineering 2004 (IEEE Cat. No.04CH37513). Vol. 1. 2004, 103–106 Vol.1 (cit. on p. 17).

[47] A. P. Reeves, R. J. Prokop, S. E. Andrews, and F. P. Kuhl. “Three-dimensional shape analysis using moments and Fourier descriptors”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10.6 (1988), pp. 937–943 (cit. on p. 17).

[48] F. Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65 6 (1958), pp. 386–408 (cit. on p. 1).

[49] S. Ruder. “An overview of gradient descent optimization algorithms”. In: (Sept. 2016) (cit. on p. 65).

[50] S. Russell and P. Norvig. “Artificial intelligence: a modern approach”. In: (2002) (cit. on p. 27).

[51] N. Schieda et al. “Diagnosis of Sarcomatoid Renal Cell Carcinoma With CT: Evaluation by Qualitative Imaging Features and Texture Analysis”. In: *American journal of roentgenology* 204 (May 2015), pp. 1013–23. DOI: `10.2214/AJR.14.13279` (cit. on p. 19).

[52] H. Shin et al. “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning”. In: *IEEE Transactions on Medical Imaging* 35 (Feb. 2016). DOI: `10.1109/TMI.2016.2528162` (cit. on p. 10).

[53] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *Computing Research Repository (CoRR)* abs/1409.1556 (2015) (cit. on p. 53).

[54] J. Smith and F. Chang. “Tools and Techniques for Color Image Retrieval”. In: *Proceedings of SPIE - The International Society for Optical Engineering* 2670 (Feb. 1996). DOI: `10.1117/12.234781` (cit. on p. 16).

[55] R. Song, T. Li, and Y. Wang. “Mammographic Classification Based on XGBoost and DCNN With Multi Features”. In: *IEEE Access* 8 (2020), pp. 75011–75021 (cit. on p. 9).

[56] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte. “A Dataset for Breast Cancer Histopathological Image Classification”. In: *IEEE Transactions on Biomedical Engineering* 63.7 (2016), pp. 1455–1462 (cit. on p. 9).

[57] M. Stone. “Cross-validatory choice and assessment of statistical predictions”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36.2 (1974), pp. 111–133 (cit. on p. 28).

[58] C. Strobl, A. Boulesteix, A. Zeileis, and T. Hothorn. “Bias in random forest variable importance measures: Illustrations, sources and a solution”. In: *BMC bioinformatics* 8.1 (2007), p. 25 (cit. on p. 27).

[59] C. Szegedy et al. “Going Deeper with Convolutions”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on p. 53).

[60] A. Tabesh et al. “Multifeature Prostate Cancer Diagnosis and Gleason Grading of Histological Images”. In: *IEEE Transactions on Medical Imaging* 26.10 (2007), pp. 1366–1378 (cit. on p. 9).

[61] A. M. Turing. “I.—Computing machinery and intelligence”. In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423 (cit. on p. 1).

[62] V. Vapnik and C. Cortes. “Support-Vector Networks”. In: *Machine Learning* 20.3 (Sept. 1995), 273–297. ISSN: 0885-6125. DOI: 10.1023/A:1022627411411 (cit. on pp. 22–24).

[63] B. Veeling. *Histopathologic Cancer Detection Dataset*. https://www.kaggle.com/c/histopathologic-cancer-detection/overview. [Online; accessed May-2020]. Nov. 2018 (cit. on p. 32).

[64] B. Veeling. *The PatchCamelyon benchmark (PCam) Dataset*. https://github.com/basveeling/pcam. [Online; accessed May-2020]. June 2018 (cit. on p. 32).

[65] B. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling. “Rotation Equivariant CNNs for Digital Pathology”. In: (June 2018). arXiv: 1806.03962 [cs.CV] (cit. on p. 32).

[66] J. Vestjens et al. “Relevant impact of central pathology review on nodal classification in individual breast cancer patients”. In: *Annals of oncology : official journal of the European Society for Medical Oncology / ESMO* 23 (Apr. 2012), pp. 2561–6. DOI: 10.1093/annonc/mds072 (cit. on p. 13).

[67] M. Weingant et al. “Ensemble prostate tumor classification in h&e whole slide imaging via stain normalization and cell density estimation”. In: *International Workshop on Machine Learning in Medical Imaging*. Springer. 2015, pp. 280–287. DOI: 10.1007/978-3-319-24888-2_34 (cit. on p. 9).

[68] M. N. Wernick, Y. Yang, J. G. Brankov, G. Yourganov, and S. C. Strother. “Machine Learning in Medical Imaging”. In: *IEEE Signal Processing Magazine* 27.4 (2010), pp. 25–38. DOI: 10.1109/MSP.2010.936730 (cit. on p. 9).

[69] N. Zhang, H. Pan, Q. Han, and X. Feng. “An Improved Classification Method Based on Random Forest for Medical Image”. In: *Advances in Intelligent Systems and Computing* 213 (Jan. 2014), pp. 21–31. DOI: 10.1007/978-3-642-37829-4_3 (cit. on p. 9).

[70] Y. Zhang and L. Wu. “An Mr Brain Images Classifier via Principal Component Analysis and Kernel Support Vector Machine”. In: *Progress In Electromagnetics Research* 130 (Jan. 2012), pp. 369–388. DOI: 10.2528/PIER12061410 (cit. on p. 9).

[71] K. Zou, J. O'Malley, and L. Mauri. "Receiver-Operating Characteristic Analysis for Evaluating Diagnostic Tests and Predictive Models". In: *Circulation* 115 (Mar. 2007), pp. 654–7. DOI: `10.1161/CIRCULATIONAHA.105.594929` (cit. on p. 36).