University of West Attica

School of Engineering

Department of Biomedical Engineering

Undergraduate Thesis

The analysis of histopathological images of colorectal cancer (CRC) using Machine Learning methods

Hapenciuc Theodora
Student ID: 48017102

Supervisor:

Dionisis Cavouras
Emeritus Professor

Aigaleo - Athens, March 2024

Πανεπιστήμιο Δυτικής Αττικής

Σχολή Μηχανικών

Τμήμα Μηχανικών Βιοϊατρικης

Πτυχιακή εργασία

Ανάλυση ιστοπαθολογικών εικόνων καρκίνου του παχέος εντέρου με μεθόδους μηχανικής μάθησης

Χάπεντσιουκ Θεοδώρα
Αριθμός Μητρώου: 48017102

Επιβλέπων:

Διονύσης Κάβουρας
Ομότιμος Καθηγητής

Αιγάλεω - Αθήνα, Μάρτιος 2024

Approved by the Defense Committee

Supervisor	Member of the examination committee	Member of the examination committee
Dionisis Cavouras	Dimitris Glotsos	Spyridon Kostopoulos
Professor Emeritus	Professor	Associate Professor

Dionysios Kavouras Digitally signed by Dionysios Kavouras Date: 2024.03.15 18:09:15 +02'00'

[illegible] Digitally signed by DIMITRIOS GKLOTSOS Date: 2024.03.15 18:18:19 +02'00'

[illegible] Digitally signed by Spyridon Kostopoulos Date: 2024.03.15 22:52:11 +02'00'

DIPLOMATIC THESIS AUTHOR'S STATEMENT

The undersigned Theodora Hapenciuc of Stelian-Ilarie, with student ID number 48017102, student of the Department of Biomedical Engineering, Faculty of Engineering of the University of West Attica, hereby declare responsibly that:
«I am the author of this thesis, and any assistance I received in its preparation is fully acknowledged and referenced in the thesis. Furthermore, any sources from which I have used data, ideas, or words, whether directly or paraphrased, are cited in their entirety, with complete references to the authors, the publishing house, or the journal, including any sources that may have been used from the Internet. Additionally, I affirm that this work has been written exclusively by me and constitutes intellectual property belonging to both me and the institution.

Violation of the above academic responsibility constitutes substantial grounds for the revocation of my diploma».

The declarant
Theodora Hapenciuc

[illegible]

Acknowledgements

First of all, I want to thank my supervisor, professor Dionisios Cavouras, for believing in me and allowing me to work and collaborate with him during this journey of writing my thesis. I appreciate his time, his guidance, and the valuable pieces of advice that he gave me, some of which went beyond the academic aspect and helped me in life. He also introduced me to Ph.D. candidate George Xenogiannopoulos, with whom I have cooperated all this time. I cannot thank George enough for helping me during this period and for his patience, support, and guidance. George was also my mentor during my internship. Because by the time I started working on my thesis, I was also starting my internship. It was a challenging period for me, but with their support, I overcame all the difficulties. I also thank my family and my friends for supporting me during all this time. latex

Theodora Hapenciuc

Abstract

With this drastic increase in cancer rates and deaths because of cancer, it is essential to find new and improved ways to prevent it as soon as possible (at early stages), especially colorectal cancer, which is ranked second in the cancers that cause death. Nowadays, it is occurring more and more among young people, often at an advanced stage. As it does not have specific symptoms, colorectal cancer is difficult to diagnose. Also, it is crucial to differentiate the benign lesions from the malignancies. Thus, a decision-support system was developed in this study to support this purpose. For this aim, a dataset of histopathological images with benign and malignant cases was utilized to generate features (68 in total), and statistical analysis was performed to find the features that can differentiate the categories. Furthermore, these features were used as input for creating a machine-learning system. Among the classifiers that we used were the K-Nearest Neighbors (KNN), Classification Trees (CART), and Random Forest. Recursive Feature Elimination (RFE) was employed for feature reduction, and the evaluation was conducted using Bootstrap and K-Fold cross-validation, focusing on accuracy, precision, and recall metrics.

Moreover, further investigation was made to differentiate the two categories by employing some Convolutional Neural Networks (CNNs) utilizing pre-trained models, such as Vgg16, MobileNetV2, ResNet50V2, InceptionV3, and EfficientNetB0. The research proceeded by splitting the two categories into five (healthy, benign, grades I, II, and III) and testing every combination in pairs of two. The same steps, as described for the two, were followed for the five categories. The top five features that were observed to distinguish the two classes were the skewness, energy range, correlation mean, LBP6, and Gabor energy range. Also, the RF classifier using the bootstrap evaluation method satisfactorily differentiated benign from malignant with an accuracy of 90.20%, 87.63% sensitivity, and 91.99% specificity. Lastly, the RES50 model showed the best accuracy of 90.92% -for 100 epochs. Our system also performed adequately for the five categories despite the reduced size of the data per class.

Keywords: CRC, benign, malignant, statistical analysis, machine learning, deep learning

Περίληψη

Με αυτή τη δραστική αύξηση των ποσοστών καρκίνου και των θανάτων εξαιτίας του καρκίνου, είναι απαραίτητο να βρεθούν νέοι και βελτιωμένοι τρόποι για την πρόληψή του το συντομότερο δυνατό (σε πρώιμα στάδια), ιδίως του καρκίνου του παχέος εντέρου, ο οποίος κατατάσσεται στη δεύτερη θέση των καρκίνων που προκαλούν θάνατο. Στις μέρες μας, εμφανίζεται όλο και περισσότερο στους νέους, συχνά σε προχωρημένο στάδιο. Καθώς δεν έχει συγκεκριμένα συμπτώματα, ο καρκίνος του παχέος εντέρου είναι δύσκολο να διαγνωστεί. Επίσης, είναι ζωτικής σημασίας να διαφοροποιούνται οι καλοήθεις αλλοιώσεις από τις κακοήθεις. Έτσι, στην παρούσα μελέτη αναπτύχθηκε ένα σύστημα υποστήριξης αποφάσεων για την επίτευξη αυτού του σκοπού. Για τον σκοπό αυτό, χρησιμοποιήθηκε ένα σύνολο δεδομένων ιστοπαθολογικών εικόνων με καλοήθεις και κακοήθεις περιπτώσεις για τη δημιουργία χαρακτηριστικών (68 συνολικά) και πραγματοποιήθηκε στατιστική ανάλυση για την εύρεση των χαρακτηριστικών που μπορούν να διαφοροποιήσουν τις κατηγορίες. Επιπλέον, αυτά τα χαρακτηριστικά χρησιμοποιήθηκαν ως είσοδος για τη δημιουργία ενός συστήματος μηχανικής μάθησης. Μεταξύ των ταξινομητών που χρησιμοποιήσαμε ήταν οι K-Nearest Neighbors (KNN), τα Classification Trees (CART) και το Random Forest. Για τη μείωση των χαρακτηριστικών χρησιμοποιήθηκε η Recursive Feature Elimination (RFE) και η αξιολόγηση πραγματοποιήθηκε με τη χρήση Bootstrap και K-Fold cross-validation, εστιάζοντας στις μετρικές παραμέτρους της ακρίβειας, ευαισθησίας και ειδικότητας.

Επιπλέον, έγινε περαιτέρω έρευνα για τη διαφοροποίηση των δύο κατηγοριών με τη χρήση ορισμένων συνελικτικών νευρωνικών δικτύων (CNN) που χρησιμοποιούν προεκπαιδευμένα μοντέλα, όπως τα Vgg16, MobileNetV2, ResNet50V2, InceptionV3 και EfficientNetB0. Η έρευνα προχώρησε με το διαχωρισμό των δύο κατηγοριών σε πέντε (υγιείς, καλοήθεις, βαθμοί I, II και III) και τη δοκιμή κάθε συνδυασμού σε ζεύγη των δύο. Για τις πέντε κατηγορίες ακολουθήθηκαν τα ίδια βήματα, όπως περιγράφηκαν για τις δύο. Τα πέντε καλύτερα χαρακτηριστικά που παρατηρήθηκαν να διακρίνουν τις δύο κατηγορίες ήταν: skewness, energy range, correlation mean, LBP6, and Gabor energy range. Επίσης, ο ταξινομητής RF χρησιμοποιώντας τη μέθοδο αξιολόγησης bootstrap διαφοροποίησε ικανοποιητικά την καλοήθη από την κακοήθη μορφή με ακρίβεια 90,20%, ευαισθησία 87,63% και ειδικότητα 91,99%. Τέλος, το μοντέλο RES50 παρουσίασε την καλύτερη ακρίβεια 90,92% -για 100 εποχές. Το σύστημά μας είχε επίσης ικανοποιητικές επιδόσεις και για τις πέντε κατηγορίες παρά το μειωμένο μέγεθος των δεδομένων ανά κατηγορία.

Keywords: CRC, καλοήθης, κακοήθης, στατιστική ανάλυση, μηχανική μάθηση, νευρωνικά δίκτυα

Contents

List of Figures

List of Tables

0.1 Abbreviations

Name	Abbreviation
CART	Classification and Regression Trees
CNN	Convolutional Neural Network
CRC	Colorectal Cancer
DL	Deep Learning
EFIB0	EfficientNet (Model B0)
FN	False Negative
FP	False Positive
GLCM	Gray-level co-occurrence matrix
GLRL	Gray Level Run Length
INCV3	Inception Version 3
KKN	K-Nearest-Neighbor
LBP	Local Binary Pattern
LDA	Linear Discriminant Analysis
LOO	Leave-One-Out
MDC	Minimum Distance Classifier
ML	Machine Learning
MLP	Multi-Layer Perceptron
MNV2	MobileNet Version 2
PCA	Principal Component Analysis
RES50	Residual Networks (Depth 50)
RF	Random Forest
RFE	Recursive Feature Elimination
STD/std	Standard Deviation
STEM	Science Technology Engineering Math
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
VGG16	Visual Geometry Group 16
WSI	Whole Slide Image

Table 1. Abbreviations across the thesis

Chapter 1

Introduction

1.1 Goal and purpose of this thesis

Nowadays, cancer is one of the primary diseases that affect human health, especially colorectal cancer (CRC), which is one of the most common cancer types, with morbidity and mortality ranking third and second, respectively, in the world as cited in [1]. It is found that the probability of getting this type of cancer is higher in economically developed societies. Moreover, it is thought of as a disease of older people, even though in the last few years, the rate of CRC in patients under 40 years old has increased immensely. It is also mentioned as a ”silent killer,” as at the early stages, it has non-specific symptoms to no symptoms at all, which can influence the diagnosis. Thus, when symptoms start to appear, the disease has often progressed, and early diagnosis, at a less advanced stage, is crucial in fighting the disease and can prolong human life. In some cases, benign (pre-cancerous) lesions can even be removed before transforming into malignant. Hence, colonoscopy is the reference benchmark for early detection, and during its procedure, benign cases can be sampled or even extracted and further examined. So, the acquired tissue is appropriately processed and examined further by pathologists. This procedure (the evaluation of the tissue and the conclusions drawn) might be complicated in some cases and even prolonged. Moreover, mistakes or misinterpretations from the doctors in detecting diseases must be considered.

However, along with the increase in the rate of cancers, we also witness promising advances in technology and diagnostic procedures. Machine learning and deep learning (domains of AI) are becoming a part of our lives and will definitely help us improve our lives in every aspect. In particular, in the medical domain, the machine learning algorithms aided in interpreting medical images and set the basis for developing CAD -Computer-aided detection/diagnosis algorithms.

Therefore, this thesis aims to analyze histopathological images of colorectal cancer using machine learning and deep learning methods. The goals that we have set are, first of all, the necessary image processing (of the histopathological images of colorectal cancer) and feature generation. Then, perform the statistical analysis of the generated features and encounter the features with a significant statistical difference between the existing categories (benign and malignant). Eventually, machine learning and deep learning systems are created to differentiate benign from malignant cases.

1.2 Methodology and Limitations

Freely available histopathological images of patients with colorectal cancer were used for the research purpose of the thesis. Firstly, patches containing the regions of interest were generated, gathered, and proceeded for feature extraction from these dataset’s images. Then, the features for the two classes, benign and malignant, were generated. These features were reduced by the statistical analysis (statistical tests) and the feature reduction method that was applied, and they constitute the input for our machine learning system. Ultimately, only essential features remain that help differentiate the two classes. Then, these patches for the two categories were used for the deep learning procedure. After that, the results of the two procedures were compared, and conclusions were drawn about the best classifiers, evaluation methods, and feature reduction methods for the ML procedure and the best network for the DL procedure.

Furthermore, the original patches for the five classes, healthy, benign/ adenomatous, grade I, II, and III, were used, and the same strategy (as for the two categories) was followed for five categories as well.

Limitations

The most crucial limitation encountered during the implementation of this thesis is that the dataset used was limited in size. In general, medical images and datasets are hard to obtain and are usually unavailable for free use and research. Also, no augmentation was applied to expand the size of our dataset (it was considered future work). In addition, the generated images were unequal; thus, there was a different number of images for every category. This imbalance between the classes impacted our process and our results. Lastly, it must be mentioned that some computational limitations were met, as the ML procedure was intensive and time-consuming in some cases (when searching for the best combination of features among numerous features).

1.3 Structure of this thesis

The following chapter of this thesis will briefly present the theory on which the research was based and established to understand definitions and concepts related to the main topic. Firstly, the second chapter presents the anatomy of the colon, then the concept of colorectal cancer (CRC), the staging of the CRC, and the screening tests to diagnose it. Then, it discusses image processing and analysis topics and provides a theoretical overview of all the features used. After that, it introduces the concept of machine learning and describes some classifiers, feature reduction methods, and evaluation methods. Also, some basic statistical concepts are introduced. Afterward, deep learning and six networks used for this thesis are discussed. Finally, a review of related work and research is presented.

The third chapter presents the tools and software utilized, the dataset's characteristics, and the methodology followed for our analysis and research. It also explains the investigation done before starting with the actual data.

In the fourth chapter, the detailed results are displayed. Firstly, the results of our investigation, followed by the results of applying the algorithm after splitting the dataset into two and five classes.

Finally, in the fifth chapter, the main findings are discussed, and the conclusions drawn from the research are presented. Also, some views and ideas for further work are mentioned.

Chapter 2

Theory

This chapter presents in detail the existing literature about the topic of the thesis and what has been done due today.

2.1 Anatomy of the colon

The digestive system breaks food and liquids into usable substances for giving energy to the body, providing growth, and tissue repair. The wastes that the body cannot use are eliminated through bowel movements. Several organs, such as the mouth, throat, stomach, small intestine, large intestine, rectum, and anus, comprise the digestive system. Additionally, the salivary glands, liver, gallbladder, and pancreas produce digestive enzymes and juices to aid food and liquids digestion. This system is also called the gastrointestinal system, and it can be seen in Figure 1.

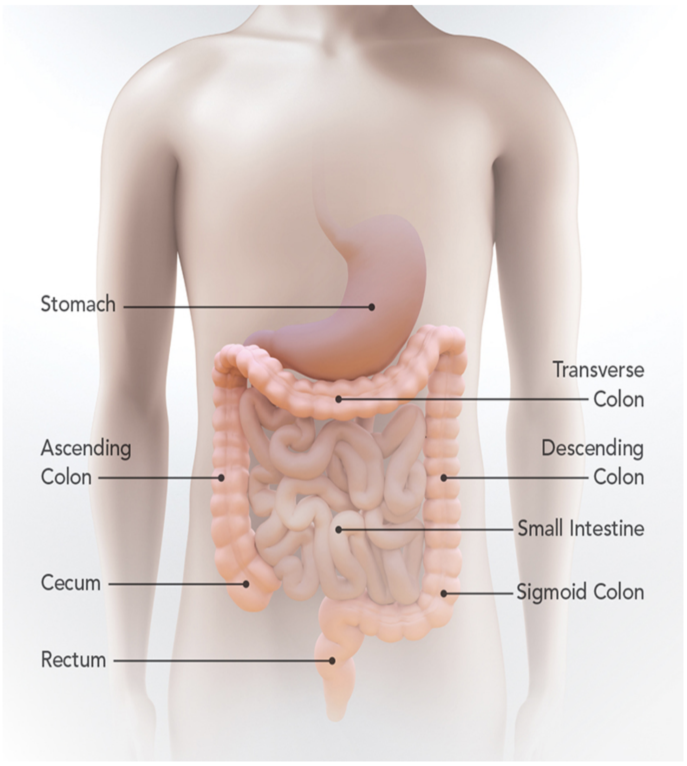


Figure 1. Shows the human's lower gastrointestinal (GI) tract. [2]

The regions or the lower part of the digestive system are the large intestine, the rectum, and the anus. Specifically, the large intestine begins with the cecum, a pouch-like form located in the right lower abdomen. The appendix, a small finger-like projection, is attached to the cecum. The colon follows the cecum, is the most extensive part of the large intestine (which measures approximately 5 feet in length), and consists of the following four parts. Firstly, the ascending colon travels up the right side of the abdomen. Then the transverse colon crosses the abdomen horizontally from right to left. Continuing, the descending colon descends on the left side of the abdomen. Finally, the sigmoid colon is an S-shaped segment that joins the descending colon to the rectum [2]. Thereafter, the rectum is found in the pelvis, slightly above the anus. Together with the anal canal, it forms the large intestine's final segment, measuring around 6-8 inches. Eventually, the anal canal ends at the anus, which serves as the large intestine's exit point from the body.

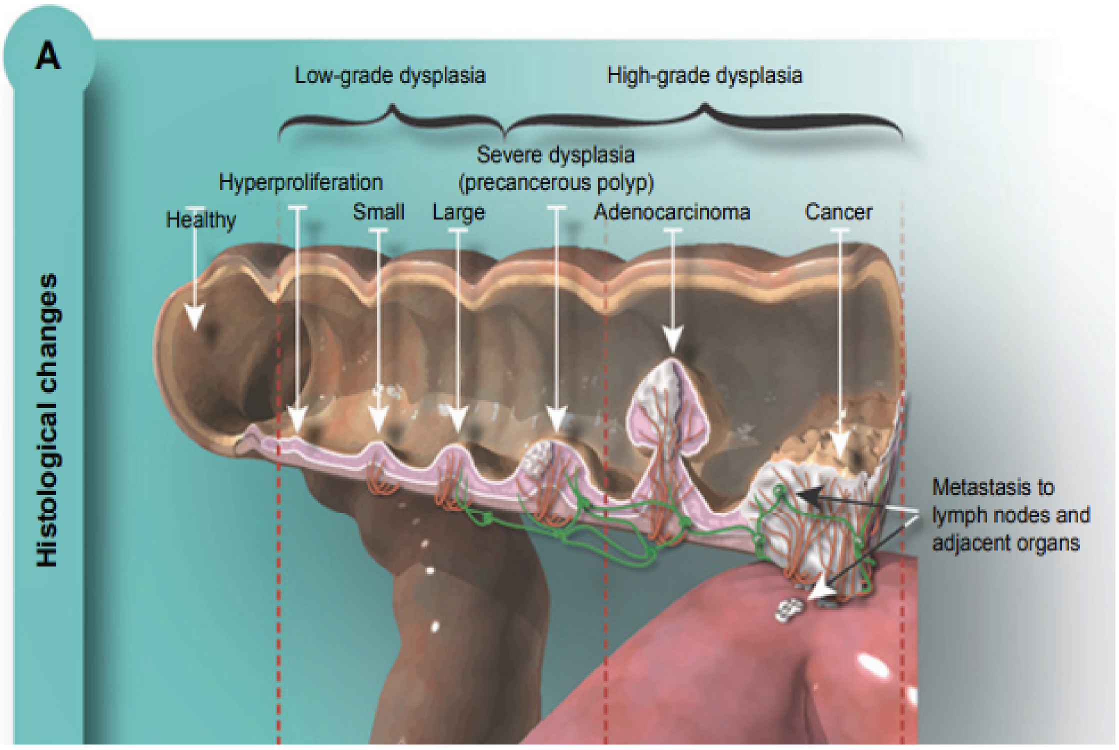


Figure 2. Large intestine showing histological changes: low-grade dysplasia, high-grade dysplasia, and metastasis [3]

The structural plan of the gut wall varies region-wise to accommodate local functional differences. Four layers constitute the mature gut wall: mucosa, submucosa, muscularis external, and serosa or adventitia [4]. The mucosa is the internal lining of certain body cavities and organs and contains mucous glands, also known as a mucous membrane. The submucosa is a layer of tissue adjacent to the mucosa. On the other hand, the outer lining of particular organs and body cavities in the chest and abdomen, such as the stomach, is known as the serosa. The lymph nodes are small, bean-shaped systems throughout the body, and they filter substances in a fluid called lymph while also helping fight infection and disease.

2.1.1 Staging of colorectal cancer

Lung cancer is the most prevalent cancer in both males and females, accounting for 12.4% of all cases. Breast cancer is the leading type of cancer in women, at 11.6% of cases. Prostate cancer is the most commonly diagnosed cancer for men, accounting for 7.3% of cases according to the World Health Association (WHO) [1]. However, colorectal cancer, or CRC for abbreviation, frequently appears in both sexes and is responsible for a high annual mortality rate considering the people dying from cancer (9.3%, which makes it second in the ranking of deadly cancers) according also to WHO [1].

The growth of CRC can affect anyone, but specific factors increase the chance of developing the disease. Some of these are diet, obesity, the absence of physical exercise, smoking, and moderate-to-heavy alcohol consumption, which are factors that can be controlled and changed. Conversely, consuming more dietary fiber, green leafy vegetables, folate, and calcium can help prevent CRC development, as mentioned in [3]. On the other hand, a family history of having colorectal polyps or CRC, genetic conditions like Lynch syndrome, personal history of inflammatory bowel disease, type 2 diabetes, and racial and ethnic backgrounds are some factors that cannot be changed.

After several research and statistics collected, it was concluded that most cases of colorectal cancer start as polyps or adenomas and then progress to cancer. Detecting colorectal cancer accurately has led to the evolution of detection at the earlier stages. However, early detection is crucial to effectively treat the patient, as only a tiny percentage of cases are diagnosed before they have progressed [3].

The growth of colorectal cancer occurs either in the colon or the rectum. It can be diagnosed during a screening test or if the patient experiences several symptoms. The patient in the early stages does not always have symptoms or has symptoms that are not immediately associated with CRC, such as abdominal discomfort, gradual weight loss, and fatigue. However, other symptoms appear in the later stages, such as blood in the stool, abdominal pain, nausea or vomiting, bowel obstruction or perforation, bloating, cramps, gas pains, and severe weight loss without apparent cause [5].

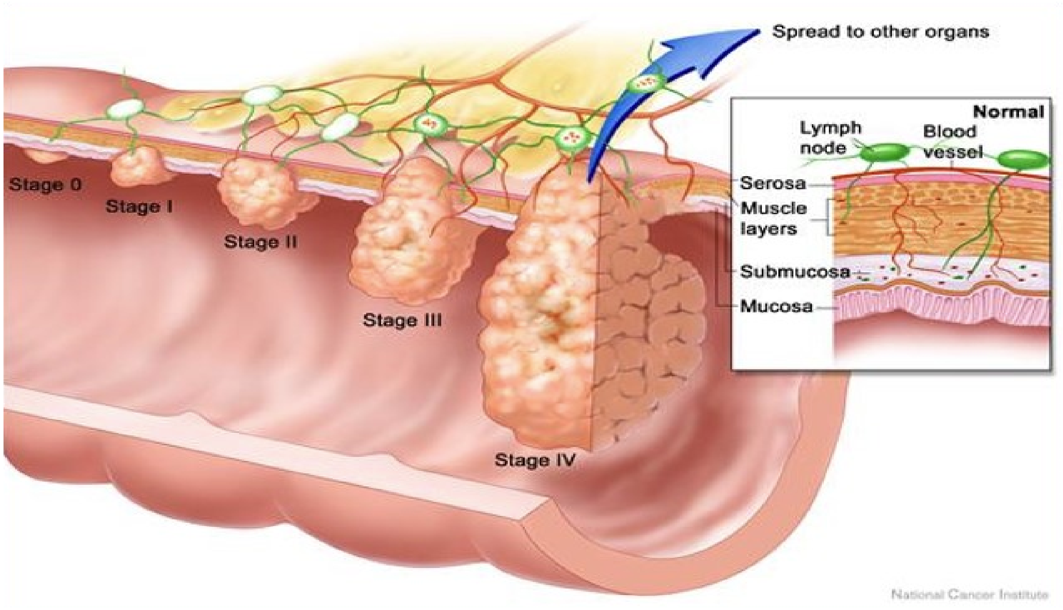


Figure 3. Stages of Colorectal cancer in the large intestine, also showing the layer of the colon (mucosa, submucosa, muscle layers, and serosa) [6]

The stages of colon cancer determine, to a considerable extent, the chance of survival, considering if the disease has advanced. Promising chances exist as long as the diagnosis is made early [5].

In the first stage, stage 0, abnormal cells are found in the colon or rectum's inner layer (mucosa). The abnormal cells can develop into cancer and spread to neighboring healthy tissues. The two main options for treatment at this stage are local tumor resection or simple polypectomy and segmental resection for more extensive lesions that are not amenable to local resection. In stage I, colorectal cancer has spread to the submucosa, which is a tissue near the mucosa or has spread to the muscular layer of the colon or rectum. The treatment at this stage is surgical resection and anastomosis.

Stage II cancer extends through the muscle layer of the skin and into the colon wall into the serosa. It then progresses to the visceral peritoneum and finally to the adjacent organs. The cancer has progressed to the outside of the colon at this stage. Treatment includes wide surgical resection and anastomosis.

Stage III is divided into three parts (A, B, C). In stage IIIA, cancer through the mucosa has spread to the submucosa or even to 1-3 adjacent lymph nodes. Continuing in stage IIIB, cancer has spread through the muscular layer of the colonic wall to the outer layer of the wall or even to the tissue surrounding the intestines and organs of the abdomen. It has affected even 4 to 6 neighboring lymph nodes. In the next stage, IIIC, cancer has spread to the inner lining of the abdomen, has affected seven (or more) neighboring lymph nodes, and has spread to tissues near these lymph nodes. The treatment prescribed in stage III cases is surgery for removing the piece of the colon containing cancer (partial colectomy) and the adjacent cancerous tissue and lymph nodes, followed by adjuvant chemotherapy.

Cancer has metastasized, in the final stage, stage IV, to other organs like the liver, lungs, ovaries, and bones. Surgery is unlikely to cure these tumors in most cases. However, if there are only a few small areas of cancer in the liver or lungs, they can be extracted surgically along with colon cancer and can extend the patient's life [5].

When it comes to rectal cancer, the innermost lining of the rectum may contain abnormal cells in stage 0, which can potentially transform into cancer and spread to nearby healthy tissue. The cancer in stage I has spread beyond the innermost lining of the rectum and affected the second and third layers, as well as the inside wall of the rectum. However, it has not yet reached the outer wall of the rectum or spread outside of it. Stage II of cancer involves the spreading of cancerous cells beyond the rectal area, although they have not yet invaded the lymph nodes. In stage III, nearby lymph nodes are affected by cancer, yet other parts of the body have not been affected, which happens in stage IV [7].

2.1.2 Screening tests and microscopes

The screening tests for CRC aim to detect abnormalities or cancer in patients who have or have not reported or observed any concerning symptoms. Are prescribed by healthcare providers for checking the individual's health after considering some high-risk factor such as age or the person's history. The screening tests are colonoscopy, sigmoidoscopy, fecal occult blood test (FOBT), decal immunochemical test (FIT), and stool DNA test. If these examinations indicate abnormalities, then additional tests, the diagnostic tests, are suggested. One standard diagnostic test is tissue biopsy during colonoscopy. Furthermore, the health care provider may suggest additional X-rays, CT, MRI, PET scans, and ultrasounds [8].

Tissue samples from the affected area are examined to obtain histopathological images of potential colorectal cancer. The procedure that is followed consists of some standard steps. First, the tissue sample, which has abnormalities, is collected from the colorectal area. This process is achieved considering one of the various biopsy techniques, such as endoscopy, colonoscopy, or surgical excision. The chosen technique is based on the peculiarities of each patient. Then, the tissue sample is instantly set in a fixative solution, usually formalin. This procedure allows the preservation of the cellular structure and stops tissue degradation. After that, the fixed tissue sample goes through a process that prepares it for microscopic examination. These steps typically include dehydration, clearing, and embedding. Dehydration involves replacing water in the tissue with alcohol, clearing removes the alcohol and replaces it with a substance that allows better visualization of the cells, and embedding involves placing the tissue in a solid medium, typically paraffin wax. Once the tissue is embedded in a paraffin block, thin slices, known as sections, are cropped utilizing a microtome (-those are usually nearly 4-6 micrometers thick). Next, the sections are placed on glass slides and subjected to various staining techniques to enhance the visualization of cellular structures and specific features. The most typical stain utilized for histopathological analysis is the hematoxylin and eosin (H&E) stain. Hematoxylin stains the nuclei blue, while eosin stains the cytoplasm pink. Finally, a pathologist (or a histotechnologist) analyzes the stained tissue sections under a light microscope like the one shown in figure 4.

They analyze the cellular morphology, tissue architecture, and any abnormal features to identify and classify the presence of abnormalities. The pathologist assesses various parameters, such as tumor type, grade, stage, invasion depth, and lymph node involvement. Also, the pathologist interprets their findings and prepares a histopathology report that documents the diagnosis and describes the characteristics of the cancer. Histopathological images of colorectal cancer are often captured using a microscope with an attached camera or a digital slide scanner. Microscopes, such as the one shown in Figure 4, have eyepieces and various objective lenses with different magnification levels, allowing the observer to examine the sample. In a camera setup, a camera port might allow light to pass through to the camera instead of or in addition to the eyepiece to capture the images. These images can be stored digitally and reviewed by healthcare professionals for further analysis, consultation, documentation, or digital image processing.

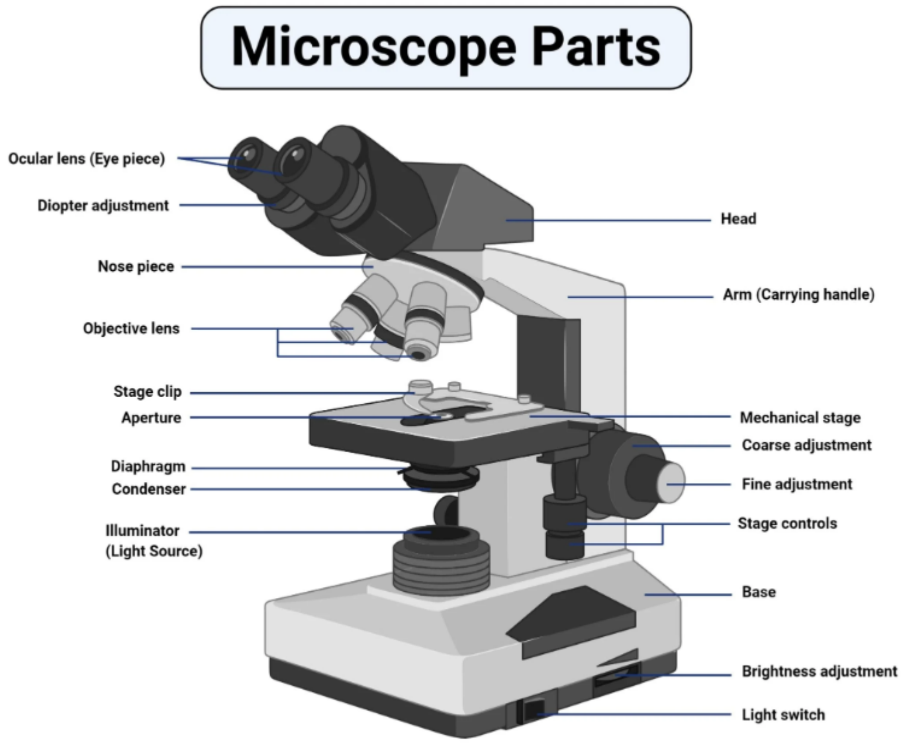


Figure 4. Parts of a microscope [9]

2.2 Image processing and analysis

Every medical image, irrespective of which acquisition interface imaging system it comes from, is a 2-dimensional digital signal, x(n1,n2). This digital signal, x(n1,n2) (in case of medical images), represents a visualization of the interior of the human body (organs, tissues), anatomy, and physiology, and it aims to help doctors diagnose diseases and various conditions that are hard to diagnose

[10]. In the computer's memory, this image is stored as a two-dimensional array of numbers called an (image) matrix. Each element of the image matrix is named a pixel and represents an amount of light in the grayscale (for black and white grayscale images). Usually, the images include a considerable number of pixels. Generally, it is preferred to maintain the storage space as small as possible; thus, the values of the pixels of an image are rounded to integers between 0 and 255 (which occupy one byte in a computer's memory) [11]. In addition, grayscale images only have a single numerical value at each pixel location. On the other hand, true color images differ from grayscale images by capturing the full spectrum of colors in a triplet vector format. This vector typically consists of RGB components and is defined at every image pixel location.

Digital image processing refers to the reversible modifications at the values of the image matrix. This phase aims to improve the image quality and extract and understand the information from the matrix. Many techniques are used for this purpose, such as increasing contrast, filtering with different filters, minimizing the noise (denoising), volume rendering, and cutting the image into pieces that include the regions of interest.

Image analysis is acquiring information from digital images. ”A picture paints a thousand words” is often encountered saying, and that is precisely the case. Most of the time, the analysis is applied after the image processing is accomplished.

Analyzing and interpreting visual data requires the utilization of different algorithms and techniques. Image analysis techniques utilize mathematical and statistical algorithms to process and analyze the pixel-level information within an image. These algorithms can extract relevant features, identify patterns or structures, measure characteristics, classify objects, and make decisions based on the analyzed information. Image analysis aims to automate the process of understanding images, helping computers to perceive and interpret visual information like humans. It contains various tasks, including image enhancement, segmentation, feature extraction, object detection and recognition, image classification, and image understanding.

2.3 Features generation/extraction

Feature extraction is obtaining higher-level information about an image from the color, shape, and texture.

As Haralick et al. mentioned in [12], it is self-evident to follow the direction toward the type of features linked to those that humans use to interpret the information coming from an image. Three essential groups of features, that humans use for interpreting image information, are textural, spectral, and contextual. The spectral features define the average tonal variations in various regions of the visible and/or infrared portion of an electromagnetic spectrum. In contrast, textural features include information about the spatial distribution of tonal variations within a region. Lastly, the contextual features include information from the examined area's region. Tone refers to the shades of gray in an image, while the texture is directed to the spatial distribution of those gray tones. Context, texture, and tone are always present in the image, even though sometimes one property can dominate the others. Also, the texture is an intrinsic property of nearly all surfaces. It includes all the knowledge about the structural configuration of surfaces and their connection to the surroundings.

Feature generation is split into two parts: the detection and the description of the features. Detection of features is about finding the characteristics of a particular image region or the image itself. Nevertheless, the description of the features is the assignment of quantitative properties in the regions where the features had been detected (that describe the valuable information of the image).

Since the 1960s, texture examination has been a subject of study, and various methods for distinguishing textures have been suggested. The technology of texture analysis is applied, most of the time, after the image processing is completed. This technology generates, by taking into consideration the distribution of image pixels, a series of quantitative textural features that are hard or even impossible for humans (doctors in our case) to identify. The way the pixels are spread in an area of the image defines the texture of this area. It is said that the texture is the image's ”signature”.

The most important and generally utilized methods that are applied to acquire textural features are statistical. These features are generated from the image histogram, the gray-level co-occurrence matrix method (GLCM), and features from the run length matrices(RLM). There are also some additional approaches, such as transform-based methods, which transform spatial information into frequency and scale information; structural methods, and model-based methods, which use sophisticated mathematical methods to describe the texture characteristics of images[13].

2.3.1 First-order statistics

The first-order statistics' features refer mostly to the frequency of pixels' gray tones that are distributed or/respectively to the numbers of the image matrices in a specific area of the image. First of all, we have the

1. **Mean value**

$$mean = \frac{\sum_i \sum_j g(i,j)}{N} \tag{2.1}$$

Where the g(i,j) is the tone of gray in the grayscale of the pixel that is located at the coordinates (i,j), and N is the number of pixels. Which gives us the mean value of the gray tone or color of the image.

2. **Variance (std^2)**

$$std^2 = \frac{\sum_i \sum_j (g(i,j) - mean)^2}{N-1} \tag{2.2}$$

Variance is the main measure of variability, and by using it, we can estimate how much the values of gray tones of the pixels are differentiated from the mean value. However, the fact that its unit of measurement is raised on the square makes it unsuitable for analysis. So that is the reason that standard deviation is used more frequently.

3. **Standard deviation (std)**

$$std = \sqrt{\frac{\sum_i \sum_j (g(i,j) - mean)^2}{N-1}} \tag{2.3}$$

Standard deviation is the square root of the variance and has the same unit of measure as the measurements.

4. **Skewness (s)**

$$s = \frac{1}{N} \frac{\sum_i \sum_j (g(i,j) - mean)^3}{std^3} \tag{2.4}$$

Skewness is a statistical parameter that determines the extent of asymmetry of a distribution (positive or negative), and it is precious when the data have outliers. If its value is around zero (0), it means that the distribution is symmetric. Negative values (negative asymmetry) mean that the number of pixels with lower gray values in the grayscale overshadows the mean value and drags the mean value to the left side of the distribution. On the other hand, respectively, positive values (positive asymmetry) mean that the number of pixels that have higher values of gray drag the mean value to the left side of the distribution, as it is noticed in the figure 5 (a) below [14].

5. **Kurtosis (k)**

$$k = \frac{1}{N}\frac{\sum_i \sum_j (g(i,j) - mean)^4}{std^4} \tag{2.5}$$

A shape characteristic that describes the sharpness of the distribution analogized to the normal distribution and estimates the comparable peak or flatness of a distribution. Values of kurtosis around zero (0) define the normal distribution, named Mesokurtic. The normal distribution, in general, is the norm/benchmark. Negative kurtosis values describe a Platykurtic distribution, and positive values represent a Leptokurtic distribution. This is also noticed in the following figure 5 (b) [14].

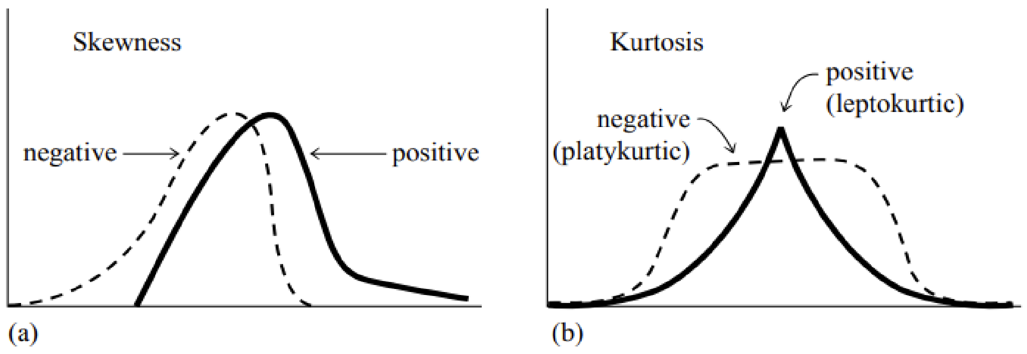


Figure 5. Visual representation of (a) Skewness (b) Kurtosis [14]

2.3.2 Second order statistics

Co-occurrence and Run Length

Second-order statistics do not derive from the original image matrix but from secondary matrices that occur from the original image matrix. The purpose of these (textural) features is to define the texture of the images. In images, textures estimate the gray level differences (contrast) and can identify regions of the images in which changes or some organized structures can be found [15]. Two groups of features occur from two different secondary image matrices: the features from the co-occurrence matrices and the features from the run length matrices. Also, it is observed that, generally, both of them (of these two groups of features) are used for image classification.

These features store information about the image's textural properties, such as its texture uniformity, linear dependencies of gray tone, the number and type of boundaries, the level of contrast, and the intricacy of the image.

Co-occurrence matrices

The co-occurrence matrix represents or describes the frequency of two/adjoining pixels coexisting in the exact area of the image, considering a specific direction (for example, 0°, 45°, 90°, or 135°), with a range of 1 (nearest neighbor). Haralick et al. [12] in 1973 was the first to present the GLCM technique, and since then, it has been excessively used, especially for medical aims (especially in the medical field). Haralick technique is divided into two phases: first, the calculation of GLMC, and then the computation of the textural features from the GLMC [16]. Actually, in the first phase, four matrices are calculated for every direction of 0°, 45°, 90°, or 135°. Several features can occur from these matrices, some of which are represented below.

We have to take into account:

$$P(i,j) \tag{2.6}$$

is the co-occurrence matrix (GLCM)

$$p(i,j) = \frac{P(i,j)}{R} \tag{2.7}$$

Is the (i,j)th entry of the normalized GLCM (which is the pixel pair normalized value of the GLCM)

$$Px(i) = \sum_{j=1}^{N} p(i,j) \tag{2.8}$$

Px(i) is an array that is calculated from the sum of the rows of the matrix, and N is the number of the different gray tones

$$Py(j) = \sum_{i=1}^{N} p(i,j) \tag{2.9}$$

Py(j) is an array that is calculated from the sum of the columns of the matrix, and also N is the number of the different gray tones

Taking into consideration the above relations, the following textural features [16]:

1. **Angular Second Moment (ASM)** which is estimating the homogeneity of the image (measures the local uniformity of the gray levels.)

$$ASM = \sum_{i}^{N} \sum_{j}^{N} p(i,j)^2 \tag{2.10}$$

2. **Energy** which also refers to the homogeneity of the image

$$Energy = \sqrt{ASM} \tag{2.11}$$

3. **Contrast** that calculates the contrast or the gray level dissimilarities between a pixel and its neighbor in a region of the image

$$CON = \sum_{n=0}^{N-1} n^2 (\sum_{i}^{N} \sum_{j}^{N} p(i,j)) \tag{2.12}$$

4. **Correlation** which estimates the linear dependency of gray level values in the co-occurrence matrix

$$COR = \frac{\sum_{i}^{N} \sum_{j}^{N} ijp(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \tag{2.13}$$

where the μ_x and μ_y are the mean values of the Px, Py, and the σ_x, σ_x are their standard deviations.

5. **Dissimilarity** gauges the distance between pairs of objects (pixels) in the region of interest.

$$Dissimilarity = \sum_{i} \sum_{j} |i-j| p(i,j) \tag{2.14}$$

6. **Homogeneity** is measuring how near the distribution of elements in the GLCM is to the diagonal of GLCM. Also, the contrast of the image lessens as homogeneity increases

$$Homogeneity = \sum_i \sum_j \frac{p(i,j)}{1+(i-j)^2} \tag{2.15}$$

7. **Entropy** is the degree of randomness or the degree of disorder in the image.

$$Entropy = -\sum_i \sum_j p(i,j) log(p(i,j)) \tag{2.16}$$

These equations apply to the occurring four co-occurrence matrices. This signifies that four values for each equation are obtained, taking into consideration the four directions as shown in Figure 6. Ultimately, we use the mean value and range of those four values as the image's features.

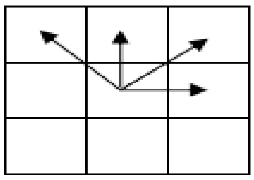


Figure 6. The directions of the co-occurrence matrices: 0°, 45°, 90°, or 135°

Run-length matrices

Chu, Sehgal, and Greenleaf published a new pair of features in 1990, as stated in [17]. These features reflect the distribution of gray levels in the run-length matrix. The run-length matrix is a two-dimensional matrix showing the count of each gray level for a specific length in the image. The run-length matrix p(i, j) is used to determine various texture features and is represented as the number of runs with pixels of gray level i and run length j for a given image.

First, we have to consider the following:

The $p(i,j)$ stands for the number of runs of pixels of gray level i and length j. The image contains M levels of gray and N distinct run lengths. Also, the total number of runs in the image is represented by n_r, and the image contains n_p pixels in total.

1. **Short Run Emphasis (SRE)**[18]

$$SRE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{p(i,j)}{j^2} \tag{2.17}$$

2. **Long Run Emphasis (LRE)**

$$LRE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) j^2 \tag{2.18}$$

3. **Gray-Level Non-Uniformity (GLNU)**

$$GLNU = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j)^2 \tag{2.19}$$

4. **Run Length Non-Uniformity (RLNU)**

$$RLNU = \frac{1}{n_r} \sum_{j=1}^{N} \sum_{i=1}^{M} p(i,j)^2 \tag{2.20}$$

5. **Run Percentage (RPC)**

$$RPC = \frac{n_r}{n_p} \tag{2.21}$$

where

$$n_r = \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) \tag{2.22}$$

$$n_p = \sum_{i=1}^{M} \sum_{j=1}^{N} jp(i,j) \tag{2.23}$$

Also, in this case, occur four (4) RL matrices, one for every direction, so four values for each feature are acquired.

2.3.3 Local Binary Pattern (LBP)

The LBP algorithm is a standard method for texture analysis combined with structural and statistical texture analysis. The basic LBP was invented during David Harwood's visit to the University of Maryland to Oulu in 1992. The initial thought was that two-dimensional textures could be defined by two complementary local measures: contrast and pattern. So, it was first used for completing measurements of local image contrast.

The first approach of the LBP algorithm (according to Topi Maenpaa and Pietikainen [19]) was taking into consideration the eight neighbors of a pixel and the value of the pixel (which was located) in the center as a threshold. Then, it multiplied the thresholded values (those above the threshold value) with some weights given to the neighbor pixels, and the final output was the sum of these numbers. The final result was substituting the central pixel's value, as shown in the figure below 7, and stored in the output LBP 2-D array. So, for each pixel of the input image, the procedure of thresholding and saving the output value in the LBP array is followed. The final phase is to calculate a histogram over the outcome LBP array, in which every one of its values will be considered a feature (this way will occur ten features).

Given that the LBP method was unaffected by small and monotonic changes in grayscale, a local contrast measure (C) was added to the algorithm. This parameter was calculated by taking the difference between the averages of the neighbor pixels' values above and below the threshold, which is also shown in the figure below 7. The local contrast and the 2-D LBP's distribution were considered features. Thus, the LBP/C operator that was occurring had excellent discrimination rates.

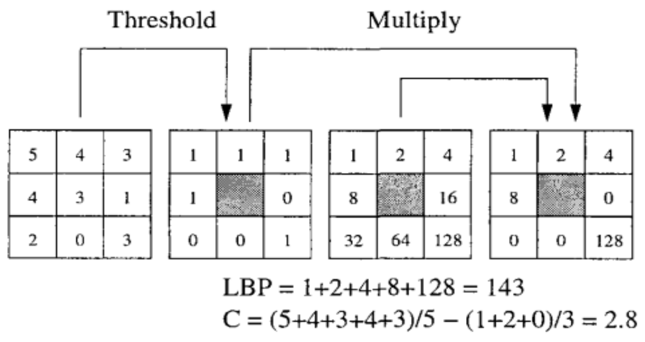


Figure 7. Visual representation of the steps of the LBP method: thresholding and multiplication with the weights [20]

The LBP method is considered a unifying approach to texture analysis. As mentioned before, the texture can be separated into the statistical and the structural approach. The statistical properties of pixels of an image are utilized to define its texture. An example of statistical texture methods is the second-order statistics arising from the co-occurrence matrices proposed by Haralick. Similarly, the LBP method can be considered a particular example of a multi-dimensional second-order statistic measure. In this approach, the texture is the input of a two-dimensional matrix, and it is interpreted using the occurring statistical parameters [21]. Also, it is pointed out that the relations between the pixels are of a local kind [22]. ”The brightness level at a point in an image is highly dependent on the brightness levels of neighboring points unless the image is simply random noise”, mentioned Cross and Jain [23]. Furthermore, Jain and Karu [24] note that texture is described not only by the gray levels of pixels but also by the local gray value ”patterns”.

The second approach, the structural, presents the idea of texture primitives. The definition of complex structures with simpler primitives is the main intention.

In general, primitive-based models have been used for clarifying the human insight of textures (Beck et al.) [25]. The LBP can also be regarded as a sample of the structural texture analysis methods. As mentioned in [21], this approach regards texture as a local pattern that is periodically repeated over an area. These two approaches can result in different findings when analyzing and classifying texture. A single approach may not be enough to detect a texture’s patterns. There is no single or proper path to analyze each texture; it always depends on the situation. The fact that a texture includes statistical and structural features, so a unified approach was proposed and used.

Rather than trying to collect information about the texture from every pixel, local patterns are created. Every pixel acquires a label, which is based on the neighbor pixels. Local primitives noticed by the LBP include spots, flat areas, edges, edge ends, curves, etc. In the following figure 8, some examples of the $LBP_{8,R}$ can be seen. In the figure below, the black circles represent the zeros, and the white circles represent the ones. In conclusion, the LBP approach is rightly considered a unifying approach that can recognize with success various textures to which individual statistical or structural methods have been applied.

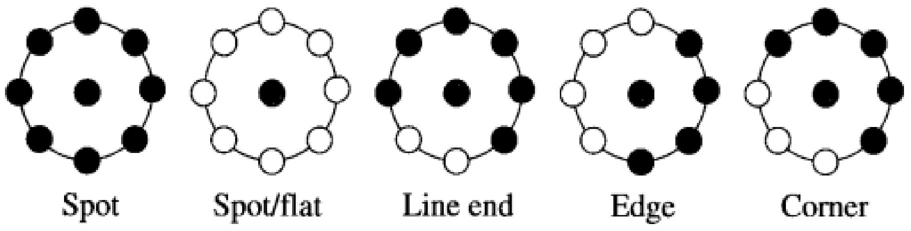


Figure 8. Various texture local patterns noticed by the LBP method [19]

As Topi Maenpaa and Pietikainen imply in [19] and Ojala et al. in [26], the LBP technique that is used today is different from the first approach, but the basic idea remains the same, where the local texture can be defined by a binary code that is constructed by utilizing the gray value of the centered (pixel) as a threshold for its neighborhood. With the first approach, capturing quite fine details in an image was possible. However, it was not feasible to notice details at different scales. So, the new extended LBP approach has to consider two more parameters: the P, which is the number of points in a circular neighborhood, and the R, which is the radius of the circular neighborhood, which permits us to consider different scales. Thus, the new extended LBP operator is symbolized as $LBP_{P,R}$. Some examples of different circular symmetric neighbor sets can be seen in figure 9.

Image texture is considered a 2-D occurrence represented by two properties: spatial structure (pattern) and contrast. Considering the grayscale and rotation invariant for texture characterization, local pattern and contrast form an intriguing pair. The local pattern is affected by rotation, but contrast is not, and on the other hand, where the grayscale influences contrast, spatial pattern is not. Therefore, contrast (C) alone is not significant, as it depends on the grayscale [26].

In conclusion, comparing the original LBP and the extended approach yields two differences. First, deriving rotation-invariant texture descriptors is more straightforward by indexing the neighborhood in the general definition circularly. Also, the neighbors' values that do not fit the pixels are calculated by interpolation.

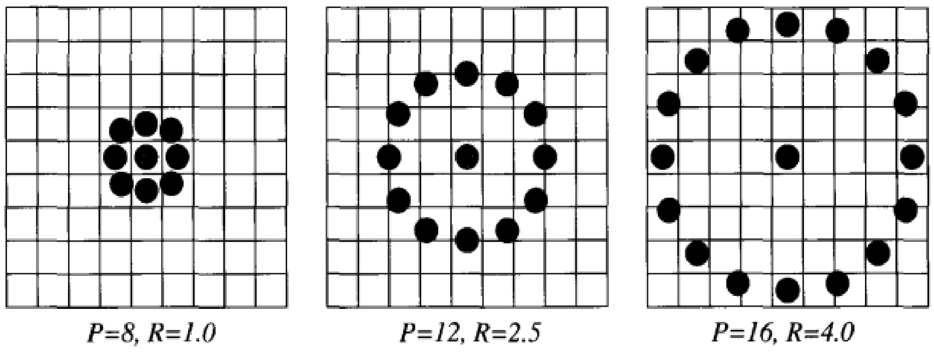


Figure 9. Three circular symmetric neighbor sets with different radius (R) and number of points (P) of the LBP method [19]

The histogram of the responses of the $LBP_{P,R}$ operator (of the "uniform" patterns) calculated for an image or an area of the image appears to be a robust and handy texture feature.

As an elementary texture operator, the LBP is ideally suited for implementations demanding rapid feature generation, and it is generally preferred due to its simplicity and performance.

2.3.4 Tamura textural features

Various types of textural features for image classification have been proposed and mentioned. After the good performance and results of Haralick's features, which basically was an easily computable approach, Tamura and his colleagues [27] were inspired to encounter new visual features corresponding to human visual perception. Notably, in this procedure, having a set of features relevant to human perception of visual textures is more suitable. The Tamura textural features approximating human visual perception are precious for utilization in feature selection. The following features were calculated and assumed based on psychological tests and experiments that had been done on humans. Six parameters, coarseness, contrast, directionality, line-likeness, regularity, and roughness, were introduced by Tamura et al., from which the first three are noteworthy. Nevertheless, all of them are essential since the last three features are highly related to the first three.

1. **Coarseness (Coarse - Fine)**

   One of the primary features occurring from Tamura's algorithm is coarseness. Sometimes, when we are referring to coarseness, we are actually indicating (in the narrow sense) texture. It is observed that in some cases, coarseness has been used as the principal clue to determine if an image includes texture or not. The greater the coarseness value is, the more irregular the texture is. When patterns are composed of different structures, the sizes of their components may indicate a difference in the scale of coarseness. So, a pattern is assumed to be coarser or, in other words, it conveys the impression of roughness when its component size is more extensive and/or its components are less repeated.

2. **Contrast (High Contrast - Low Contrast)**

   Contrast measures the way gray levels vary in the image, a measure that can be acquired by the statistical analysis of the gray level distribution of its pixels. So, actually, by having an image with patterns that vary only in their gray-level distribution, we can differentiate the brightest from the darkest patterns of the image more efficiently, and contrast can be measured. Some factors that may impact the contrast of two different patterns with different textures can be the gray-level range dynamic, the ratio of the black and white in the image, the frequency of intensity changes, and the sharpness of edges. Contrast, in a more limited sense, can represent the quality of the image. Also, it must be noted that higher contrast indicates a smaller variety of gray levels distributed along the image, which in some cases can imply loss of information. In conclusion, higher contrast of patterns with different structures does not always mean better diversification of those patterns. The optimum level of contrast must be found so that useful information is not lost.

3. **Directionality (Directional - Nondirectional)**

   Likewise, an additional global property in an image area is direction. Directionality involves the shape of the components and the placement rule. Some images have an orientation, several orientations, or no orientation at all. However, only the total degree of directionality is measured, as Tamura et at. state in [27]; the orientation does not matter. Thus, patterns differing only in orientation have the exact extent of directionality.

4. **Line-likeness (Line-Like versus Blob-Like)**

   This feature is interested only in the shape of the texture elements. It provides additional information to the information we obtain from the three previous essential features, mainly when directionality cannot be used to differentiate two patterns. By using the definition of line-likeness, we describe a component of texture that is formed of lines. Therefore, when the direction and the adjoining (edge's) direction for a given edge are almost equal, we consider such a group of edge points a line.

5. **Regularity (Regular - Irregular)**

   For natural features, it is not that easy to estimate a degree of irregularity; details about the size and the shape of the primitives must be known. Assume that if any feature of a texture varies over the whole image, the image is irregular. So, regularity is associated with (the abovementioned features) coarseness, contrast, directionality, and line-likeness. Also, it is calculated as a sum of the standard deviation of the previous four features.

   $F_{reg} = 1 - r(\sigma_{crs} + \sigma_{con} + \sigma_{dir} + \sigma_{lin})$

   where r is a normalizing factor and each σ_{xxx} stands for standard deviation of the corresponding feature.

6. **Roughness (Rough - Smooth)**

   Initially, this feature was not used for visual textures but only for tactile ones. Nevertheless, when examining natural textures, we can compare them in terms of roughness or smoothness. Emphasis had been given to calculating the coarseness and contrast. Hence, roughness can be estimated as a measure that occurs by totaling the coarseness and contrast.

   $F_{rgh} = F_{crs} + F_{con}$

Generally, as mentioned earlier, the first three of Tamura's features are used in most situations. They are beneficial (they can come in handy) in image analysis, and they can capture high-level perceptual characteristics of a texture.

2.3.5 Wavelet

The wavelet transform, as mentioned in [28], like the Fourier transform, pertains to the class of integral transforms and can be seen as an addendum to the Fourier decomposition method. In a few words, the sinusoids utilized in Fourier analysis are large waves, whereas, a wavelet represents a small wave, which practically is an oscillation that decays fast. The scalability of wavelets is the essential difference between the wavelet transform and the Fourier transform.

So, the classical Fourier transform analysis in the time-frequency analysis of a signal was insufficient because the Fourier transform of a signal did not include any regional information. The windowed Fourier transform (a short-time Fourier transform known later as the Gabor transform) was then presented by Dennis Gabor in 1946 to conquer this liability. After that, Meyer [29] discovered the existing literature on wavelets. And, thereupon many outstanding mathematicians made a noteworthy contribution to the wavelet theory. Thus, the wavelet transform is advantageous when investigating and analyzing signals and images with considerable discontinuities and it provides powerful and flexible filters to analyze images at different scales.

As it is mentioned in [30], noteworthy information is not always found in low- frequency areas of a signal or an image(s). Therefore, a more suitable approach, to using the wavelet transform for the detection of the textures, is to find the important frequencies and then decompose the images/signal further.

The primary step in all the wavelet transforms is convolving the signal (or image) with a filter bank to acquire helpful information about the signal/image. By applying the discrete wavelet transform (DWT) as shown in figure 10 (a), the image is decomposed; specifically, it is separated into four sub-bands. Every sub-band filter yields four image regions with half of the side length. The image regions are organized by agreement, as displayed in figure 10 (a). These sub-band images are named as follows: LH1 (Low-High; the number 1 indicates the level of decomposition), HL1 (High-Low), HH1 (High-High), and LL1(Low-Low). So, as it is mentioned in [31], an image's texture can be described using these sub-bands images, their values, or the features that can be extracted from them. To achieve the second level decomposition, the LL1 sub-band is further decomposed and critically sampled, as shown in figure 10 (b). Likewise, LL2 is utilized for additional decomposition, and so on. This procedure will continue until the wanted(significant) scale of frequency is achieved (getting, in our case, the essential information about the texture of the images).

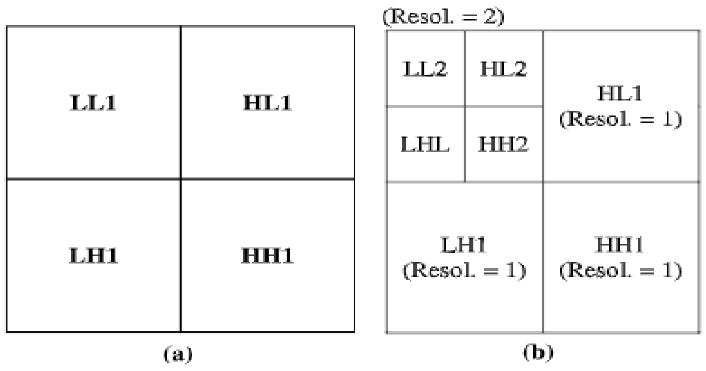


Figure 10. Two levels of decomposition after using the discrete wavelet transformation: (a) The traditional pyramid-type first level (b) Second level [31]

2.3.6 Gabor features (Transform-based method)

The Gabor filter, described in 1946 and named after Dennis Gabor, is a convolutional filter representing a combination of Gaussian and a sinusoidal term. It is generally used for generating features that represent the texture and edges in image processing and image analysis procedures. It has been found that Gabor filters are analogous to those of the human visual system. Also, Gabor filters are widely used for texture feature extraction and are considered a powerful tool for texture analysis based on signal processing.

In 1986, Turner [32] was the first to use a bank of Gabor filters to analyze texture. To extract frequency and information for the orientation, this bank of filters at diverse scales and orientations was used, permitting multichannel filtering of an image. So, this bank could then be used to disintegrate the image into texture features [33]. The implementation that is followed is inspired by the one suggested by Manjunath et al. [34]. The features' statistics, computed by filtering the image with a bank of orientation and scale-sensitive filters in a region, are utilized to describe the underlying texture information and details.

The Gabor function is described as follows, according to Fogel and Sagi in[35]:

$$G(x,y|\lambda,\theta,\psi,\sigma,\gamma) = exp(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2})exp(i(2\pi\frac{x'}{\lambda}+\psi)) \tag{2.24}$$

where,

$$x' = xcos\theta + ysin\theta \tag{2.25}$$

$$y' = -xsin\theta + ycos\theta \tag{2.26}$$

and λ represents the wavelength(or the frequency), the θ the orientation of the filter, the ψ describes the phase offset, the σ the standard deviation and γ symbolizes the spatial aspect ratio [35].

Let's suppose the I(x, y) is the input element matrix and $G(x,y|\lambda,\theta,\psi,\sigma,\gamma)$ the Gabor operator. We can calculate the G*I spectra for various orientations and wavelengths. This range identifies the texture element.

Two features occur for every combination of orientation and wavelength: energy and amplitude. In our approach, the mean and the range of these two features for five different directions and frequencies/wavelengths have been calculated.

2.4 Machine Learning

Before discussing machine learning, we need to explain what a model is and what is its connection to machine learning. A model is just a mathematical representation of the relation between different variables. For example, a cooking recipe includes a model that correlates inputs like the number of servings with the amount of every required ingredient [36]. The goal is to develop models by using known data (training phase) and then to be able to predict different results regarding a problem or an issue using new/unknown data. So basically, **machine learning**, as Burkov is referring in [37], which is a subfield of Artificial intelligence, can be described as the procedure of solving a practical problem or extracting knowledge and information by collecting data and algorithmically assembling a model based on that data. Eventually, this model is used to solve the practical problem or to acquire information.

Types of Machine learning (ML)

1. **Supervised learning**

   In this type of ML, the training data given to the algorithm include their labels. Thus, training a model on labeled data is the essence of supervised learning. Labeled data provide input features and their respective output labels. The goal is to learn and predict the correct output label for new input data. Supervised learning consists of two types of procedures. In the classification process, the first of the two procedures, the output variable, is categorical, and the model assigns input data to predefined classes or categories. In the regression procedure, the output variable is continuous, and the model indicates a numerical value based on the input features [20].

2. **Unsupervised learning**

   In unsupervised learning, the training data has no label (e.g., Clustering). Unsupervised learning aims to identify concealed patterns, structures, or associations in the data without explicit knowledge of the output labels. It can be arranged into two categories: clustering and dimensionality reduction. Firstly, clustering involves grouping similar data points based on their intrinsic characteristics without any prior information about the class labels. Secondly, dimensionality reduction methods aim to decrease the number of input features while retaining the most suitable information. This can help visualize high-dimensional data or improve the effectiveness of subsequent machine-learning tasks.

3. **Semi-supervised learning**

   Lies between supervised and unsupervised learning. It addresses scenarios where a limited amount of labeled data and more data without a label are available for training: the model leverages labeled and labelless data to enhance performance in semi-supervised learning. The idea is that the data without labels can provide additional information or patterns to assist in better generalization and decision-making.

4. **Reinforcement learning**

   It involves an agent that learns by interacting with an environment to earn a reward. The agent's actions in the environment are met with prizes or punishments. Developing an optimal procedure that instructs the agent to make the proper choices in various situations is typically the objective. Also, it is generally utilized in various fields, such as game playing, robotics, self-driving cars, and autonomous systems.

5. **Batch and online learning**

Batch learning, also known as offline learning or batch training, involves training a machine learning model on a whole available dataset at once at regular intervals such as weekly, monthly, etc. In batch learning, the model is trained to utilize all the available labeled data in a single iteration. The model learns the relationships and the patterns in the data to generate predictions or classifications.

Online learning, also known as incremental learning or streaming learning, is a different approach where the model is updated incrementally as new data arrives. In online learning, the model learns from individual instances or smallish batches of data, one at a time, rather than using the entire dataset at once. Online learning is beneficial when data is arriving continuously, and there is a need for real-time learning and adaptability to changing patterns. It is more memory and computationally efficient than batch learning since it processes data incrementally.

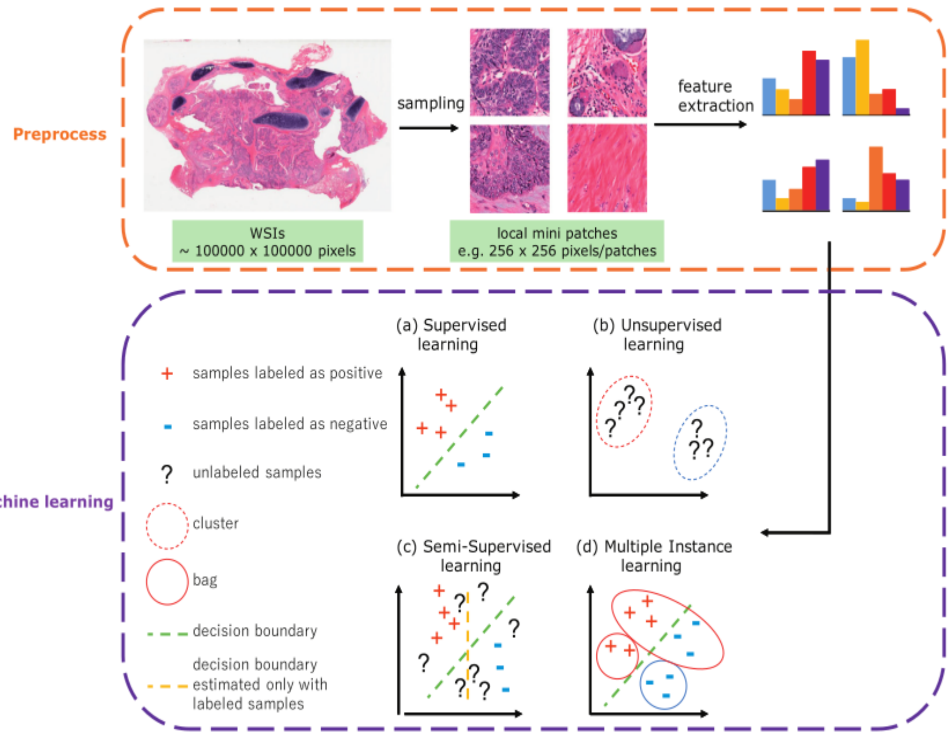

Preprocess
sampling
feature extraction
WSIs
~ 100000 x 100000 pixels
local mini patches
e.g. 256 x 256 pixels/patches
Machine learning
samples labeled as positive
samples labeled as negative
unlabeled samples
cluster
bag
decision boundary
decision boundary estimated only with labeled samples
(a) Supervised learning
(b) Unsupervised learning
(c) Semi-Supervised learning
(d) Multiple Instance learning

Figure 11. Machine learning steps: Step-1. pre-processing of the WSI and feature generation, Step-2. utilization of the ML algorithm [38] [39]

In Figure 11 above are displayed the typical steps for machine learning in digital histopathological image analysis, which show that different types of machine learning algorithms (like the ones noted above) could be employed after the Whole Slide Image (WSI) is preprocessed [38]. This preprocessing procedure might include extracting local patches in various sizes (256 x 256, 96 x 96, etc.) sampled from the large histopathological images. Then, multiple features are generated from these patches, which will be the input for the ML process, where the ultimate goal is to predict the class to which a new sample belongs [38]. Then, feature extraction and classification between benign and malignant cancer are performed for each local patch. Feature extraction aims to extract valuable information for machine learning tasks.

It is also necessary to comment that in a machine-learning system, two problems may occur and must be considered so that everything will work smoothly. These problems are overfitting and underfitting. Overfitting, shown in Figure 12 on the right, is a situation in which the resultant model fits too well with the training data and does not have good generalization capability, thus providing low accuracy when used on new data [40]. On the other hand, underfitting, displayed in Figure 12 on the left, is a situation where the model does not fit well, even on the training data. To overcome the underfitting problem, the feature selection should be increased. Sometimes it is even recommended to choose another model.

Thus, finding the best/"magic" number of features to be utilized as input for every model is crucial. An example of a proper fit is shown in the middle of Figure 12.

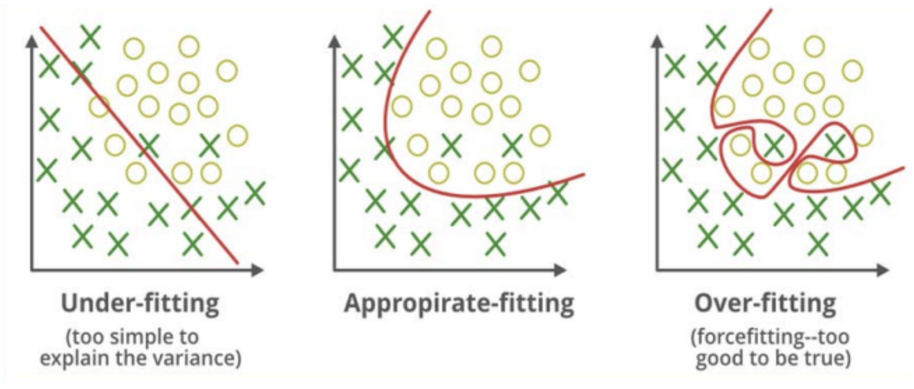

Under-fitting
(too simple to explain the variance)
Appropirate-fitting
Over-fitting
(forcefitting--too good to be true)

Figure 12. Visual representation of underfitting and overfitting [41]

2.4.1 Classifiers

Classifiers are algorithms that have been designed to categorize a vector of features into one of the Ni (i=1,2,3..) categories. The features go via a procedure of classification after their extraction. So, practically, classifiers are a family of algorithms, which are just mathematical representations, as has been mentioned before, and can separate the samples into different categories. The features and the number of different classes are the inputs that are provided to the classifiers. Also, the output of the classifiers is usually a number that indicates the class that the sample(s) was categorized.

In supervised machine learning, there are two different types of (supervised) classifiers: the statistical or parametric classifiers and the non-statistical or non-parametric classifiers. The statistical classifiers prerequisite statistical information and the a priori probability the samples (for classification) have to belong to one of the classes. Two of the most common parametric classifiers are Bayes and Minimum Distance Classifier classifiers. On the other hand, the non-statistical classifiers do not require previous training or any statistical information for the unknown data or/and the probability of these data belonging to one of the classes.

1. Minimum Distance Classifier (Nearest Centroid)

   Among several classifiers that can be found today, the minimum distance classifier (MDC) is one of the first, actually, and one of the most simple and economical ones. The MDC is a statistical classifier that uses all the available patterns from all the classes for training, and its purpose is to correctly label as many unknown samples as possible. So, how does it work? The MDC algorithm finds the centroid of every class and uses it as the only criterion for classifying the test pattern. It measures the distance of the test/unknown pattern from each class's centroid, and then the test/unknown pattern is classified into the class with the nearest centroid to it. In other words, it is selecting the class with the closest centroid to the test pattern.

   In consequence, MDC can classify efficiently without the need for additional parameters besides the centroid of each class or specific settings, and it requires the minimum computational need. However, all these might negatively impact the accuracy. Also, many pattern classification applications, e.g., for disease diagnostics, are taking advantage of the MDC, as mentioned in [42].

2. K-Neighbors Classifier

   The k-neighbors classifier is also one of the most straightforward and really practical classifiers. It is a non-parametric classifier, and it is used widely in multiple applications since it does not demand statistical information or previous training. It is also considered a lazy algorithm that performs without a training stage but has an expensive testing phase, as noted in [43]. Every sample is given as an input to the classifier, and it is compared to the previous instances, and it is classified into the class where the k nearest patterns belong, with k usually representing an odd number (k=1, 3, 5...). There is no general rule for calculating the distance between unknown and the already classified patterns. However, there are several approaches that have a different impact on the classification results. A challenging part is specifying the neighborhood, discovering the right k, and determining how the distance between instances can be measured. [44]. Some of the proposed mathematical types for calculating the distance are given beneath:

   (a) Euclidean Distance

   $$d_e(a,b) = \sqrt{\sum_{i=1}^{N}(b_i - a_i)^2} \tag{2.27}$$

   It Occurs from the Pythagorean theorem and measures the distance between two patterns a, b in a space of N dimensions (hence N characteristics).

   (b) City Block Distance

   $$d_{cb}(a,b) = \sum_{i=1}^{N}|b_i - a_i| \tag{2.28}$$

   It was used instead of the Euclidean distance formula to reduce the time of calculating distances as it is observed that the squaring of distances does not affect the result.

   (c) Maximum Distance

   $$d_m(a,b) = max_{i=1:N}|b_i - a_i| \tag{2.29}$$

   An extreme case where it is considered only the distance of the pattern that shows the biggest difference.

   (d) Minkowski Distance

   $$d_r(a,b) = [\sum_{i=1}^{N}|b_i - a_i|^r]^{1/r} \tag{2.30}$$

   A generalization of the three previous distance formulas where r is a parameter that determines the r-normal distance.

3. Gaussian Naive Bayes

   For machine learning, Naive Bayes is one of the most useful and valuable classifiers. It is a probabilistic parametric classifier based on the independence assumption, i.e. assumes that each feature is different and independent of the others. Even though, in real life, this assumption is seldom considered. As a parametric classifier, however, it presupposes the existence of at least an indicative (a small) number of samples for training. The following formula defines the generalized format of the discriminant function of the classifier:

$$d_i = ln(P_i) - \frac{1}{2} ln|C_i| - \frac{1}{2}((x - m_i)^T C_i^{-1}(x - m_i)) \tag{2.31}$$

where C_i is the covariance matrix of class i, m_i is the mean of the features of the training samples of class i, x is the test/unknown pattern to be classified. The P_i represents the possibility that this pattern belongs to class i [45].

4. Linear Discriminant Analysis (LDA)

   An additional supervised machine learning algorithm, LDA, is utilized for classification purposes. Its primary purpose is to find a linear combination of the available features that maximizes the separation between two or more categories in the dataset [46]. Instead of utilizing one feature at a time, considering a combination of two or more features can be more beneficial and contribute to the classification of the samples. A specific feature cannot be used to decide where a sample belongs, but combining it with another feature may determine the class it belongs to.

   The discriminant analysis calculates functions from a group of known data (training data) for which the observations class is known. The functions can then be applied to the testing data and predict to which class they belong. The produced discriminant functions are calculated depending on the number of categories available. For k classification groups, k-1 functions are produced. A discriminant function has the following format:

$$LD = w_1 * x_1 + w_2 * x_2 + w_i * x_i \tag{2.32}$$

   where LD is the result of the linear discrimination, w_i are the rescaled weights, and x_i are the features used for the discrimination. These weights are estimated using the separation function :

$$Separation = \frac{Between\ group\ variance}{Within\ group\ variance} \tag{2.33}$$

   The goal is to transform the data so that the ratio of between-class variance to the within-class variance, shown above in equation (2.33), will be maximized.

   The first step of the LD's training phase is to calculate every class's mean and scatter matrix. The scatter matrices measure the data variability within each class. The next step is to calculate the mean and scatter matrix again, but this time, consider the data across all classes.

   The total separation is calculated as follows $S = S_W^{(-1)} S_B$ where S_W^{-1} is the (pooled) within-group scatter matrix, and S_B is the between-group scatter matrix.

   As noted in [47], the eigenvectors and eigenvalues of the S scatter matrix are estimated, reducing the data's dimensionality while retaining the most essential information for classification. The eigenvectors with the highest eigenvalues correspond to the most important directions that maximize the data separation between the classes and are used to transform the samples onto the new subspace. Now, the data can be projected in eigenvectors' directions to create a lower-dimensional representation of the data, which is optimal for classification. The transformed data for the two classes problem is shown below in Figure 13. Once the linear discriminant function and weights have been calculated, the new samples can be projected onto the discrimination line and determine to which class they belong.

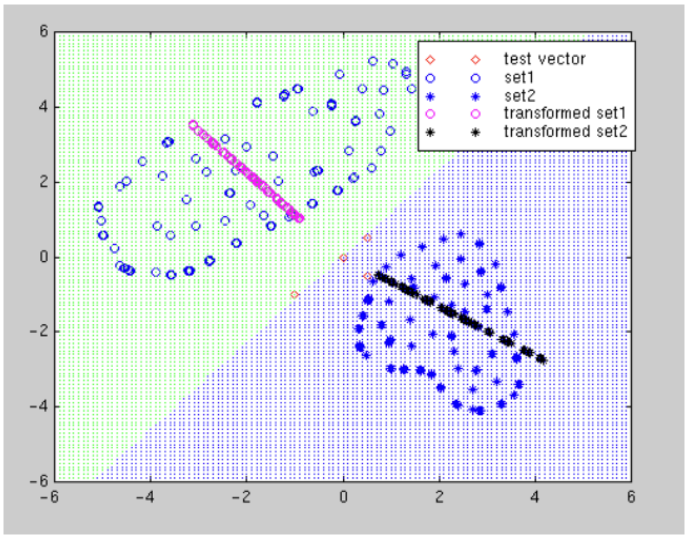


Figure 13. Separation of the data using the LDA classifier [47]

Concluding, LDA is a famous algorithm for classification studies because it is simple and efficient. It is also useful in some exceptional cases when the number of samples is small in comparison to the number of features, which can lead to overfitting when using other classification algorithms. Moreover, it has the advantage of providing interpretable results, which can help understand the connections between the calculated features.

5. Logistic Regression

Foremost, it needs to be clarified that logistic regression is a classification algorithm, not a regression [37]. Logistic regression estimates the possibilities for classification problems with two possible outcomes [48]. In addition, logistic regression belongs to the parametric category and, like the KNN, tries to indicate the class to which an unknown sample belongs. Moreover, it attempts to compute the probability of class membership [49].

In logistic regression, it is still needed to model y_i as a linear function of x_i (inputs), yet, with a binary y_i (output). The linear combination of features such as $wx_i + b$ is a function that travels from minus infinity to plus infinity, while the output y_i has only two possible values [37].

The logistic function is used to compress the outcome of a linear equation between 0 and 1 and is defined as follows:

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)} \tag{2.34}$$

As shown in Figure 14, the logistic function serves the classification problems. The output of f(x), can be interpreted as the probability (odds) of y_i being positive, having the appropriate values of x and b. For example, if the output is higher than or equal to the threshold class of x, is positive; differently, it is negative. The selection of the threshold differs depending on the problem.

The logistic regression model looks as demonstrated below:

$$f_{w,b}(x) = \frac{1}{1 + e^{-(wx+b)}} \tag{2.35}$$

where w is a D-dimension vector of parameters and b is a real number. The notation $f_{w,b}$ means that the model f has two parameters, the w and b values.

The term (wx + b) is seen from linear regression. Also, to find the best values for the w and b parameters in logistic regression, the Maximum Likelihood Estimation (MLE) is used to maximize the likelihood of observing the given binary outcomes under the logistic regression model [37]. Generally, in statistics, the MLE method is utilized to estimate one model's parameters. The regression models become ineffective when the connection between the outcome and features is nonlinear or when features interrelate [48]. All things considered, logistic regression is a fairly straightforward parametric method, and it is employed in clinical audits [50].

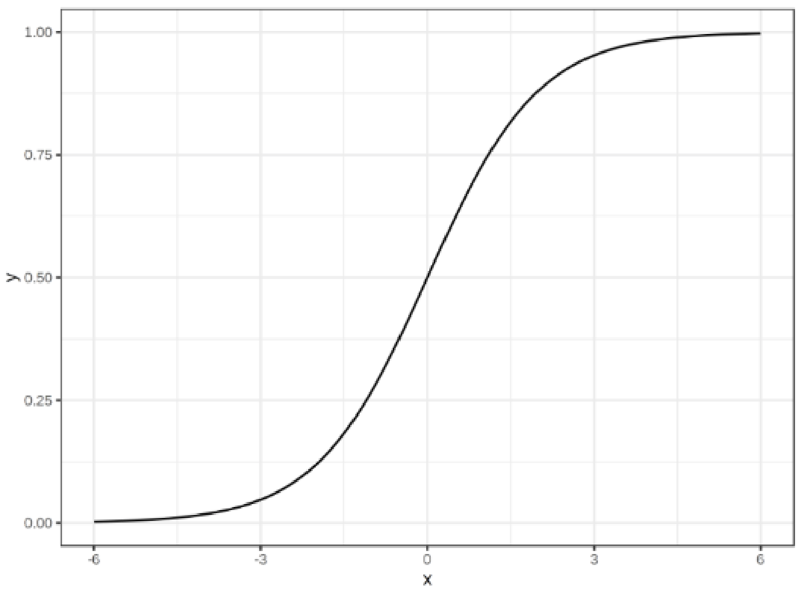


Figure 14. Visual representation of the standard logistic function (diagram) [48]

6. Perceptron (Basic version)

Many scientists and researchers developed models based on discoveries from biological neural research to comprehend how the human brain works. Back in the days of 1960s, Frank Rosenblatt designed a machine named the perceptron that performed like a human mind based on these first models. Even though it did not have the brainpower or the ability to think itself, it could learn, and this was a significant innovation that set the starting point for the development of neural network technologies.

Thus, a perceptron is a network that achieves the imitation of associative memory. The most fundamental perceptron architecture is formed of two layers of nodes that are strongly connected between them, the input layer and the output layer, and each connection is given a modifiable weight. Further, the network takes a set of inputs and is able to compute the desired output. The ability to modify and adapt the weights as needed to yield a specific output represents how the network is trained so that it eventually could learn. It seems that perceptrons are among the most premature and essential models of artificial neural networks that are still involved (and preferred) in many applications [51].

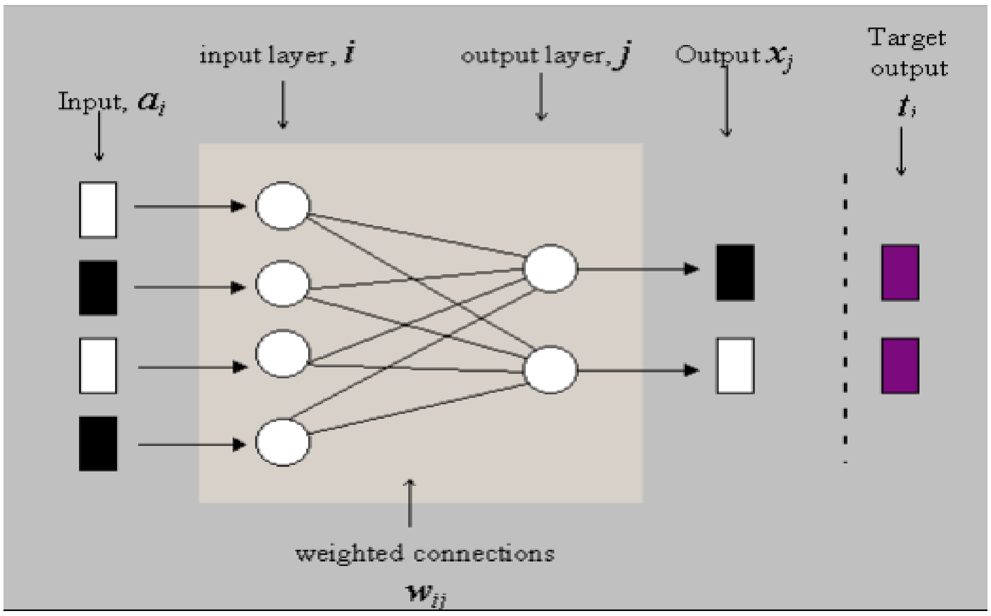


Figure 15. A simple perceptron diagram showing the [51]

To describe how it works, we are taking into consideration the fundamental architecture that contains two layers, where each layer is entirely connected to the other and does not exist any connections between nodes within the same layer, as shown in Figure 15. The first layer, the input, transmits a signal to the second layer, the output. Then, the corresponding weights of the connections are applied to the signal, and this value is obtained and summarized by the node of the second layer. The node releases an output signal if the sum reaches a provided threshold. Lastly, the outputs are totaled from all the inputs a node acquires in the output layer.

7. Multi-Layer Perceptron (MLP Classifier)

   An MLP neural network works like the perceptron classifier mentioned above, with the difference that it contains one or more hidden layers (of computation nodes) among the input and the output layers [52]. Connections are only allowed in nodes of successive levels, not within the same layer, and every node of a layer connects to all the nodes in the layer above it.

   The training phase is equal to choosing proper weights for all the connections such that a preferred output is rendered for a specific input. For MLP, the procedure of learning and the speed at which it is learning is essential.

8. Support Vector Machines (SVMs)

   SVM is a robust classifier that performs well on various classification problems. One of SVM's essential strengths is handling high-dimensional data, which can be utilized effectively in the case of datasets with many features. SVM is a general-purpose classifier that can be applied to linear or non-linear, separable data, with or without overlapping between the classes [53].

   In the case of linearly separable and non-overlapping classes, the training phase aims to find two parallel hyperplanes. The criteria to be satisfied are that neither of the data samples is located between those two parallel hyperplanes and the distance between them is maximized. The area between those two parallel hyperplanes is called the margin, and the decision boundary is defined by the hyperplane located at an equal distance between them. This is shown in the Figure 16 below:

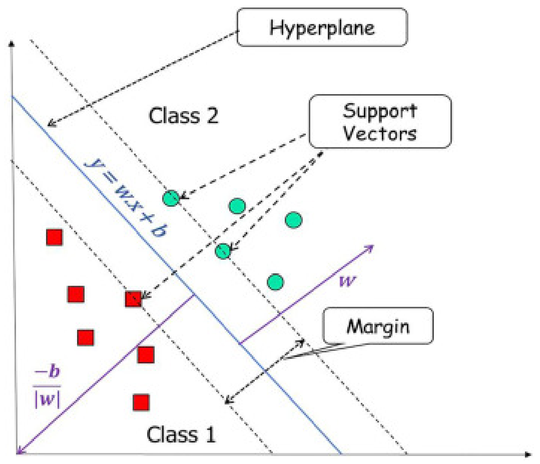


Figure 16. Two linearly separable classes with the hyperplane and the parallel to it support vectors [54]

Where the hyperplane is $wx + b = 0$ and the support vectors are $wx + b = \pm 1$, where +1 refers to class 2 and -1 to class 1, w is the vertical vector to the hyperplane, and b is the threshold that describes the distance of the decision hyperplane from the beginning of the axes which is equal to $\frac{b}{|w|}$.

When the classes cannot be separated linearly, the issue is to transform the training data from the original feature space to a new higher-dimensional space in which the classes might be separable. In the case of three dimensions, the decision boundary will be a plane, and in more than three dimensions will be a higher-dimensional surface.

SVM transforms the input data into a higher-dimensional space using a kernel function to find the optimal hyperplane. The kernel function measures the similarity between two data points in the higher-dimensional space, making it possible to find a linear decision boundary that separates the data. Some of the most commonly used Kernel functions are: linear kernel, polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel.

In conclusion, SVM is a universal and robust classifier that can be used with many datasets and handle high-dimensional data. Nevertheless, it can be computationally intensive and may demand attentive tuning of its parameters to acquire the best outcomes. Another weakness of SVM is that no probability of class membership is given; the classification outcome is cleanly dichotomous [49].

9. Decision Tree Classifier

As noted in [48], the decision tree model, also known as CART (Classification and Regression Trees), can be used successfully in cases where the association between the features and the output is nonlinear or in the event where the features are related to each other, whereas, the logistic regression model performs badly (in those cases).

A decision tree model uses a tree structure to describe a set of potential decision paths and an output for each path. The main structure of a decision tree model contains the following components: nodes, leaves, and branches. The foremost node is called the root and is presented at the top of the structure. The branches and leaves are located below the root [37]. The branches deriving from the root define the potential outcomes of a decision. Each branching node corresponds to a feature, and the end of each branch (decision path) that cannot be separated anymore is the leaf node (decision). These nodes and branches work together to make predictions.

The creation of the tree model comprises the training stage, and the important phases of its creation are the splitting phase, the stopping, and the pruning phase [55].

As the tree grows, it is necessary to find and utilize the appropriate features and conditions for splitting the data into categories based on the values of the (input) training data. Then, it is also crucial to understand when it is best to stop. It is meaningful to discover the proper balance between overfitting and underfitting. Pruning will be required when the tree grows extensively (considering many features and conditions). Some branches that do not impact the model's accuracy are removed/truncated to keep the decision tree simple and manageable.

Each branching node (as mentioned above) represents a data feature when building a decision tree, as presented in Figure 17. The sample compares a specific feature F (of the feature vector) to a specific threshold. If the value (of the instance) is above the threshold, the sample follows the branch that satisfies the condition (in our case, if it is above the threshold, it follows the right branch); otherwise, it follows the other branch. This process is repeated until the model reaches the leaf node. At this point, it is decided on the class to which the sample belongs [37].

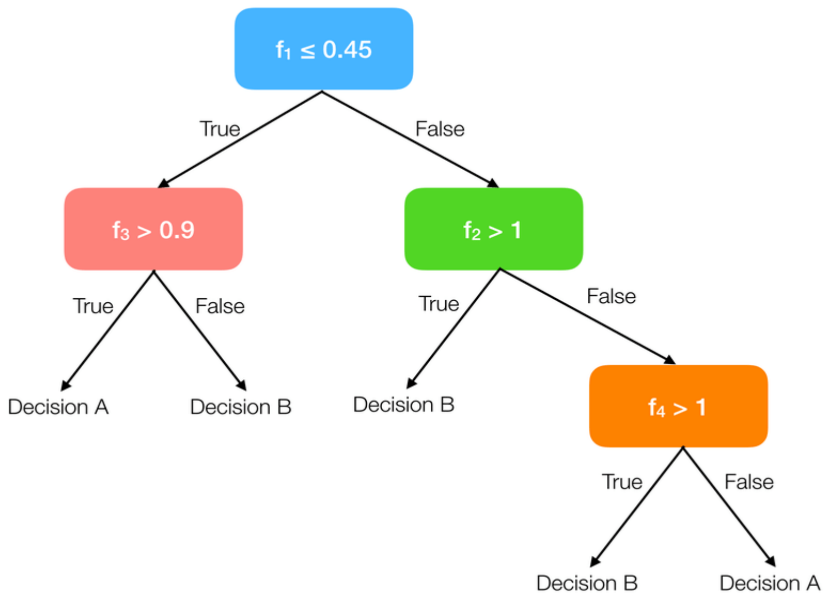


Figure 17. A decision tree classifier using and displaying four different features [56]

Unlike the other machine learning algorithms mentioned above, the decision tree method has the benefit that it is not a mysterious (black-box) model. Still, it can efficiently be represented as rules and imitate the way humans think using this "if/else-then" format, as noted in [49]. Also, the features' importance is evident, and the conditions can be seen without much effort. On the other hand, it does not perform well when having linear associations between the data and the features. Moreover, they can be a little inconsistent because a totally different tree can be produced by modifying some of the training data.

10. Random Forest Classifier

A random forest is a classifier containing a group of N decision tree classifiers, like the ones mentioned above. Each tree votes for the most suitable class for a given sample [57]. This is a continuation of the basic CART method.

The training data are separated into subgroups, and each subgroup is used as input for developing a decision tree model (classifier). For every subgroup of the training data, a different decision tree is developed; this way, a set of trees is formed, the random forest. Every tree makes a prediction for every new sample **x** (votes for the most suitable class). The calculated majority of the votes (of these predictions) is considered the final decision as to where the sample belongs.

After the training phase of this classifier, there are N decision trees. The prediction for a new sample x is acquired as the average of the votes in regression problems:

$$f(x) = \frac{1}{N}\sum_{n=0}^{N} f_n(x) \tag{2.36}$$

Where $f_n(x)$ represents the vote given from the tree n $\in$ N for the instance/input x $\in$ X, showing the probability (P(C=cj|X=x)) of x to be distributed to the class cj $\in$ C and the majority vote in classification problems, as cited in [56].

Some of the random forest's advantages are the multiple decision trees which can help to mark/ discover a wider range of patterns and relationships in the data. Moreover, using a different subgroup of the training data in the training stage can lessen the correlation between the trees, contributing to reducing overfitting. Lastly, the random forest can manage high-dimensional data and missing values.

2.4.2 Feature Reduction Methods

Before the feature reduction, a pre-process must be followed so that the importance of the features can be estimated correctly. This pre-process stage includes data normalization (and rescaling), outlier removal, handling missing data, etc.

The feature reduction phase is very important when creating a classification system, considering that, most of the time, a wide variety of features are available. It is essential that the excessive number of features will be reduced.

The goal is to choose the most essential features so that their number will decrease and simultaneously maintain all the possible information they have on separating the classes. Features that provide low distinctiveness between the classes lead to poor class discrimination. On the other hand, with the proper feature selection, the classification process becomes more efficient.

This means that the values of the features must differ a lot between different classes, but they should be alike within the same class.

1. Significance Test ranking

   When selecting features, it is essential to examine each one individually and assess its ability to differentiate samples at a given problem.

   Significance test ranking is a feature reduction method that uses statistical significance tests to select the most relevant features for a classification problem. The concept is to test the null assumption that the feature is unrelated to the class label and reject this hypothesis if the p-value is below a particular threshold.

   Some significance tests that can be used are ANOVA and t-test. The process of significance test ranking typically implicates the following steps. First, the p-value for each feature is calculated using a significance test. Then, the features based on their p-values are ranked, with the smallest p-value showing the most relevant feature. The last step is to select a subset with the most relevant features. A simple way is choosing the top k features of all features with a p-value below a certain threshold.

The significance test ranking method is simple to implement. It can be helpful when the number of features is extensive, and the goal is to select the most relevant features for the classification task. Yet, it is essential to mention that the method makes the assumption that the features are independent, which may not be accurate in practice. While this approach is not the most efficient, it allows the elimination of the obviously wrong choices quickly and avoids computational overburdening by using more complex feature reduction techniques.

2. Correlation ranking

   Correlation ranking is a feature reduction method that identifies and removes highly correlated features from a dataset. It is based on the understanding that highly correlated features can introduce redundancy in the data and potentially lead to overfitting in classification model training. The correlation between two features is a statistical measure that quantifies their relationship.

   The correlation can be positive, negative, or zero/neutral. A zero correlation indicates no correlation between the features. The positive correlation is marked when two (or more features) tend to change in the same direction. On the other hand, the negative correlation indicates that the features are not changing in the same direction, that one is increasing and the other is decreasing. The correlation coefficient between each pair of features in the dataset is calculated. The Pearson method calculates the correlation coefficients when having normal data distribution [58]. The absolute values of these correlation coefficients are then used to rank the features in descending order of correlation strength. Features with higher absolute correlation coefficients above a specified threshold are considered more correlated and potentially redundant, so they are removed.

   It's important to note that correlation ranking should be used cautiously, as the high correlation between features does not necessarily imply causation or indicate the importance of a feature for the classification task. It is always recommended to consider the specific requirements of the classification problem before applying correlation ranking [59].

3. Mixed Criterion (correlation and ranking test)

   Mixed criterion is a feature reduction method that combines significance test ranking and correlation-based methods. It is a way to combine the strengths of both methods to obtain a more robust feature selection.

   The mixed criterion method considers both the statistical significance of the relationship between the feature and the class label (significance test) and the strength of the relationship (correlation coefficient). This can help avoid the problem of selecting irrelevant features, which can happen when using significance tests or correlation-based methods alone.

4. Principal Component Analysis (PCA) ranking

   PCA is a technique that helps reduce the complexity of high-dimensional data by identifying the essential information and transforming it into a smaller set of non-correlated variables known as principal components. This approach enables us to keep the essential features of the original data while reducing its dimensionality. The idea is to find and show the data in lower dimensions (the resulting principal components), dimensions that can capture the maximum variance in the dataset.

The PCA method follows some phases to create the final principal components. Firstly, it is crucial to normalize the values of the features (data) so that features with larger values (outliers) do not influence the analysis. Secondly, the covariance matrix of the normalized data/features is calculated, which provides details about how each feature changes. Then, the eigenvectors and eigenvalues are computed by decomposing the covariance matrix. Eigenvectors represent the directions along which the data varies the most, while eigenvalues represent the amount of variance described by each eigenvector. After that, the eigenvalues are sorted in decreasing order, effectively representing decreasing variance in the data. The first-k eigenvectors are selected as the principal components, where k is the desired number of dimensions or the amount of variance explained. The principal components are created by multiplying the normalized data with the selected eigenvectors. Finally, the original feature values are projected on the new feature space described by the selected principal components. Through this transformation, the data is represented in a lower dimension, and the dimensions are not correlated with each other [60].

PCA allows dimensionality reduction while preserving the essential information of the data. It is easy to execute and has low computational requirements.

However, the user needs to define the number of principal components, which can negatively impact the analysis. It is also worth noting that PCA is a linear transformation technique and may not capture complicated nonlinear relationships in the data. Nonlinear extensions of PCA, such as kernel PCA, can be used to handle such cases [60].

5. Recursive Feature Elimination (RFE)

   Feature reduction algorithms commonly yield two types of outputs: a sorted list or a feature subset. The feature sorting technique is a greedy algorithm that generates a sorted list of features. This method can obtain the optimal subset and adjust the number of features as needed. The RFE method is a model-based reverse search algorithm that follows a greedy approach. It starts with all features and progressively eliminates the least essential until a desired number of features is reached.

   The process starts by choosing a machine learning model and training it on the entire group of features. Then, the features are ranked or evaluated for their importance. The least essential features are eliminated, and the model is retrained on the reduced feature group. This process is repeated until the wished number of features is achieved or a stopping criterion is fulfilled.

   RFE helps to identify the most essential features by considering their impact on the model's performance. By eliminating less critical features, RFE can improve the model's efficiency and reduce overfitting. It's important to mention that the choice of the stopping criterion, such as the desired number of features or a predefined threshold, is crucial in RFE, and the user must select it cautiously [61].

2.4.3 Evaluation Methods

The final stage in designing a machine learning system is the evaluation of the classifiers, calculating the accuracy of the classification results, and the probability of error. It is crucial to evaluate the performance of the classifiers on the testing data and on new, unknown data to see their effectiveness. Also, by evaluating the classifiers, possible problems can occur (such as overfitting and underfitting). Catching and identifying these problems can help improve the classifiers by changing hyperparameters or even using another algorithm. So, the evaluation phase contributes to the comparison of different classifiers on a specific problem. This can be done by comparing their outcomes on the same dataset and eventually selecting the one with the best results.

The evaluation process is performed using a group of a specific and different dataset from that used in the training process. For this reason, the available data should be used appropriately for both the training and testing phases. Some commonly used methods for evaluating classifiers are described below.

1. Self-consistency (Resubstitution method)

   This method uses all the available data samples to train the classifier (just once, the first time), and then the classifier is tested using these same data. This method reaches a relatively high classification success rate for this data set, but the success rate on the unknown is decreased (compared to the rate of success of the known data). This is because the classifier learns this particular data, and no conclusion can be drawn about its behavior on unknown data. By mathematical analysis of the method and performing the critical reviews on results using normal distributions, it is shown that the approximation of the accuracy of the classifier is proportional to the ratio of the number of samples to the number of features. In particular, the results are more accurate, considering a significant number of samples and fewer features [62].

2. Hold out

   The available data set is randomly divided into two parts: One section is reserved for training the classifier and the other for its evaluation. An advantage of this evaluation method is that the classification is done with data that have yet to take part in the training and are therefore considered unknown to the classifier. However, the method has two disadvantages: First, the separation of data reduces the number of samples used for training and evaluating the classifier (the classification system), and second, it is not (always) straightforward which criteria should be followed for separating the data and which is the optimal size of the two data segments (the two sections, one for training and one for testing/evaluating). It has been found that the larger the number of train samples, the lower the probability of misclassification. However, this way, the number of samples used for evaluating the classification system is reduced. Despite the studies and efforts being made, there is still no ideal way of separating the available data [62].

3. Leave-one-out

   For a set of N standards, this method proposes using (N-1) samples for training and then classifying the ”leftover” sample. The process is repeated N times so that each sample is classified once (by itself), and in this manner, the system's overall accuracy is calculated. This way, the independence of the training from the testing data is acquired. At the same time, there is no concern about finding the optimal separation point and, in general, about handling the samples, problems that exist in the hold-out method. The Leave-One-Out method overcomes the disadvantages of the two previous evaluation methods (self-consistency and hold-out) but requires significantly more computational time and power. With this method, the system's accuracy results seem to have a lower success rate than the hold-out or self-consistency methods. However, they are more reliable and representative when generalizing with the classification of unknown data [62].

4. K-fold cross-validation

   This evaluation method was developed to manage the large number of operations required by the leave-one-out method and is essentially a generalization of it. In the case of cross-validation, a set of N available samples (of the data set) is randomly divided into K equal parts/regions containing k samples, each one of these subgroups. For example [k1,k2, ..., kK] where obviously $K * k = N$. The classifier is trained each time with (K-1) subgroups from the following subsets (k1, k2, ... kK), and in the end, classifies the subset not used in training. The process is repeated K times so that each one of the K subgroups will be used for testing/evaluating (the training process) once. As mentioned above, this method drastically reduces the number of operations and, thus, the high level of computation needed while maintaining the statistical independence of the samples and, therefore, the reliability of the results [62].

5. Bootstrap

   The bootstrap evaluation method involves randomly selecting data instances from the original dataset with replacements to create a new dataset (the bootstrap dataset). In all the previous evaluation methods, every instance used for developing the training dataset was used only once. In the bootstrap evaluation method, one instance of the dataset, after being selected once for creating a (bootstrap) training dataset, is placed back into the original dataset and can be selected again. The probability of an instance chosen for the bootstrap dataset is $p = 1 - (1 - \frac{1}{N})^N$ and if N is a really large dataset, then the $p = 1 - e^{(-1)}$. This is why this method is called sampling with replacement. The process of creating new bootstrap datasets is repeated multiple times. The average performance metric is calculated over all the runs by training and testing the model on each bootstrap dataset. This metric accurately estimates the model's actual performance on unseen data. The bootstrap method is particularly useful when the original dataset is small [62].

2.4.4 Useful statistic concepts

Confusion matrix

Is a tabular expression that outlines the performance of a classification model. It is constructed based on the comparison between the predicted labels and the actual labels of a set of samples.

	Classified at malignant	Classified at benign
Has malignancy (malignant)	TP (True Positive)	FN (False Negative)
No malignancy (benign)	FP (False Positive)	TN (True Negative)

Table 2. Confusion matrix definition

The confusion matrix yields several useful statistical measurements, such as sensitivity, specificity, and overall accuracy.

Sensitivity and specificity are two additional performance metrics often used in binary classification problems. Sensitivity seen in (2.37), also known as recall or true positive rate, measures the proportion of true positive predictions out of all actual positive samples. It suggests how well the model can correctly identify positive instances.

$$Sensitivity = \frac{TP}{TP + FN} \tag{2.37}$$

In simpler terms, sensitivity indicates the percentage of actual positive samples that the model correctly identifies.

On the other hand, specificity presented in (2.38) measures the proportion of true negative predictions out of all actual negative samples. It implies how well the model can correctly identify negative instances.

$$Specificity = \frac{TN}{TN + FP} \tag{2.38}$$

In simpler terms, specificity indicates the percentage of actual negative samples the model correctly identifies. Sensitivity and specificity provide complementary information about the model's performance. Both metrics are necessary depending on the specific context and goals of the classification problem. For instance, in medical diagnostics, sensitivity may be more critical to minimize false negatives so that a positive for a malignancy case will not be missed and the diagnosis and treatment delayed -which might be fatal.

Overall accuracy is typically defined as the ratio of correctly predicted samples (true positives and true negatives) to the total number of samples. The overall accuracy is calculated from the following formula:

$$OverallAccuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.39}$$

By considering sensitivity, specificity, and overall accuracy together, it is better comprehended how the model performs in different aspects of classification accuracy, which helps make informed decisions based on the specific requirements of the problem at hand [63].

Statistics (short introduction)

The word ”Statistics” seems to come from the Latin word ”status,” which means state and was first used to describe a nation's population. Alternatively, it is said that it might derive from the ancient Greek word $\sigma\tau\alpha\tau\iota\zeta\omega$, which means sort/classify/categorize. It is noted that the statistics were used from ancient times. Even though statistics were less developed than today, they were important; there was a need to keep demographic records to achieve the proper functioning of the state.

Nowadays, statistics have an immense role in our lives, and they can be found and applied to every knowledge domain, such as social and human sciences, history, philology, economics, STEM, business administration, and health sciences.

R. A. Fisher, the father of modern Statistics, in [64] defines statistics as a set of principles and methodologies on how:

1. how to design the technique for collecting the data
2. how to briefly present them
3. how to analyze them and extract conclusions

Useful statistic terms

Population: In statistics, population refers to an aggregate of all individuals or items defined on some common characteristics.

Variable: is a characteristic often but not always quantitatively estimated, containing two or more values or categories that can vary from person to person, place to place, and time to time. The values of a variable constitute the ”data.” There are various types of variables: qualitative, quantitative, nominal, ordinal, discrete, and continuous. Data is the plural form of datum. It is the raw material of statistics. Data stands for a collection of numbers or facts used as a basis for making a conclusion. In other words, data are the raw, unorganized facts and figures collected from any field of inquiry.

Probability distribution: A function in mathematics that assigns probabilities to different possible outcomes in an experiment is known as a probability distribution. In simpler terms, a probability distribution estimates the likelihood of each outcome occurring.

Normal distribution: In normal distribution, the data are symmetrically distributed around the mean value, and most values are gathered around a central region, with values tapering off as they go further away from the center. The mean, mode, and median (measures of central tendency) are identical in a normal distribution.

Frequency: is the number of times a variable is repeated in the population.

Frequency distribution: The premise of data in frequency distribution describes the basic pattern the data assumes in the group. Frequency distribution provides a better picture of the data pattern if the number of items is large enough.

Null hypothesis: A hypothesis to be tested for acceptance or rejection is termed a null hypothesis. H_O denotes it.

Alternative hypothesis: is a statement regarding the population parameter or parameters, which provides an alternative to the null hypothesis H_0 within the range of pertinent values of the parameter, i.e., if H_0 is accepted, what hypothesis is to be rejected and vice versa. H_1 or H_A denotes the alternative hypothesis [65].

Statistical techniques

The statistical techniques can be classified into the following categories: descriptive and inferential statistics. The goal of **descriptive statistics** is the straightforward and efficient presentation of data. This presentation shall be made using appropriate numerical and graphical methods so that conclusions can be drawn directly for the sample set and the whole population from which the sample set was drawn. Descriptive statistical analysis is usually the first phase of statistical research and processing. The first stage of descriptive statistical analysis is to calculate various metrics (e.g., mean, median, variance) meant to describe the data so that the researcher can acquire insight into these data by summarizing them into appropriate measures. The second stage includes the design of appropriate graphs to represent the data. These graphs represent one variable at a time or the analysis of several variables so that the researcher can investigate the existence of relations between the variables. Some examples of those graphs used are the histogram, the boxplot, and the scatter plot.

Histograms, tables, and other graphs can generally present a frequency distribution. A frequency distribution displays how often a particular value is found in the whole dataset and is used for both qualitative and quantitative data. It shows the number, percentage, or frequency of various results that appear in a given dataset. Each entry in the graph or table is accompanied by the number of times the value appears in a particular range or group. These tables or graphs provide a structured way of presenting a summary of grouped data which is classified based on mutually exclusive categories and the frequency of occurrence in each respective category.

Histograms are a way of visually checking whether or not a variable approaches the normal distribution. This is particularly important considering that one of the main assumptions of many statistical methods is the normality of the data, i.e., that the data come from the normal distribution. The normal distribution resembles a bell curve and is thought to be the most commonly found in nature. Many methods a researcher can use are based on the normal distribution.

If the data is derived from a normal distribution, the histogram is expected to follow a bell-shaped pattern and appear symmetrical. It should be clear that it is unlikely to be a fully symmetric histogram for reasons of randomness. However, if the data comes from a normal distribution, there will be a strong indication of symmetry. The histogram is constructed as follows: On the horizontal axis of a system of orthogonal axes, the boundaries of the classes are marked in an appropriate scale. Consecutive rectangles are then constructed, each with a base equal to the class's width and a height such that the rectangle's area is equal to that class's frequency.

The **boxplot** is a graph that shows a general overview of the data as it allows us to observe the location and variability of the data; it can also help in making comparisons between subgroups and populations and identify the existence of outlying observations. It is sometimes called a five-point diagram as it gives us information about the mean or median, the two quartiles, and the minimum and maximum values. Therefore, from a boxplot, we can see:

- Information about the location (where the median is approximately) and the variability. The larger the box and the larger the whiskers, the greater the variability.
- Some information about asymmetry. If the data distribution is symmetric, the line inside the box should be approximately in the middle. Also, the length of the whiskers should be similar.

Three boxplots are shown below in Figure 18 representing the weight of some individuals. The horizontal line in the center of every graph corresponds to the median. The box that frames it extends from the first to the third quadrant. The whiskers, starting from the edge of the box, extend to the minimum and maximum value, respectively. The whiskers reveal variability outside the upper and lower quartiles. However, when some observations are considered outliers, they are represented with symbols instead of lines. The * (star) stands for an extreme value (or a very extreme value). In Figure 18, can be observed some extreme values in models 1 and 3.

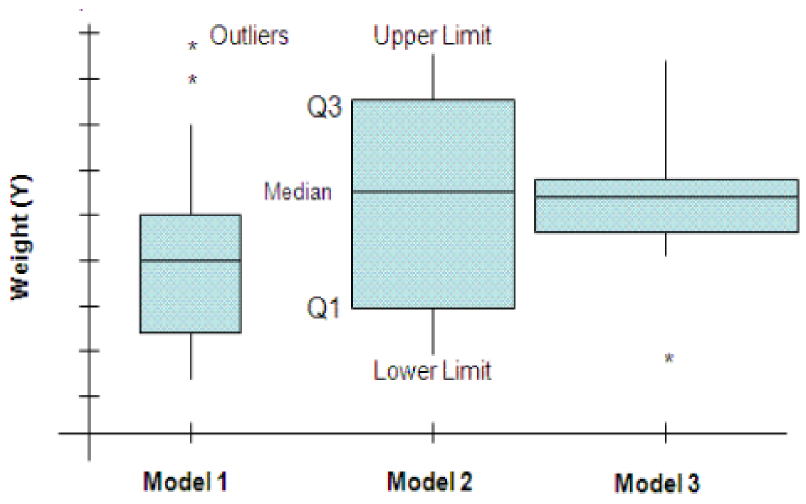


Figure 18. Three boxplots examples showing how data of a population are distributed [66]

Moreover, the **scatter plots** are graphs indicating a relationship between variables. The graph is easily made by plotting each observation as a point in a two-dimensional diagram, the coordinates of it being the values of the observation for two variables that correspond to the axes. The most important conclusion that can be obtained is the presence of a connection between two variables.

Inferential statistics is a process by which logical conclusions about a population are made based on results from a sample set drawn from that population [67]. The steps in data-driven decision-making are the following: firstly, formulating the hypothesis, then finding the proper test, executing the test, and lastly, making a decision. Hypothesis testing, an essential type of inferential statistics, compares entire populations or assesses relationships between variables using sample sets. If the statistical hypothesis determines the distribution, it is called a simple hypothesis. Otherwise, it is named a composite hypothesis. Commonly, a hypothesis is specified beforehand—before applying the statistical test that will help to make a decision. The term hypothesis means the presentation of a parameter's postulated or stipulated values. Also, some connection between parameters is hypothesized in the circumstance of two or more populations. Hypotheses are tested utilizing statistical tests to draw valid inferences. Therefore, based on observational data, a test is performed to determine whether the postulated hypothesis is accepted. This involves a certain amount of risk, termed **the significance level**. The significance level is represented by ”α”, and is often determined as 0,05 or 0,01. Level $\alpha = 0,01$ is used for high precision and $\alpha = 0,05$ for moderate precision. It is the measure of how tolerable the possibility of denying the null hypothesis (H_0) when it is true [67].

After applying a test, a decision is taken about accepting or rejecting the null hypothesis (or the alternative hypothesis). There is always some possibility of making an error in deciding about the hypotheses. These errors can be of two types. Type I error: Reject null hypothesis (H_0) when true. Type II error: Accept null hypothesis (H_0) when false.

There is also the p-value concept that is noteworthy to be mentioned. The p-value is the probability of rejecting or failing to reject the null hypothesis (H_0). H_0 is usually the hypothesis that two groups have no difference for a specific variable. The ”p” in the p-value stands for probability that two groups have no difference [68].

Thus, the marginal value of the level of significance α, for which the hypothesis H_0 is found to be rejected based on a random sample x, is called the p-value of the test. It is helpful to provide the p-value of a statistical test, particularly when H_0 is rejected because it indicates the degree of dissimilarity between the null hypothesis and the data. It is typical to say that there is very strong evidence against H_0 when the p-value of the test is ≤ 0.01. When $0.01 < p \leq 0.05$, it is said that there is strong evidence against H_0, while when $0.05 < p \leq 0.10$, it is said that there is enough evidence against H_0. Very rarely is a null hypothesis with a p-value greater than 0.10 rejected. The sample size strongly influences the p-value. Thus, p-value has become a crucial aspect of data-driven decision-making in biomedical research. The primary method employed to detect statistically significant differences amongst groups of observations is the extended null hypothesis significance testing. This approach involves computing a single p-value from sample data, which is then compared against a threshold (the α), e.g., of 0.05, to evaluate the evidence against the hypothesis of non-significant differences among groups or the null hypothesis. When applied to datasets with a large sample size, the estimated p-value tends to decrease, rendering the null hypothesis meaningless in such situations [69].

Concluding: The measurements of central tendency and dispersion, like mean, median, mode, mean deviation, standard deviation, etc., and the frequency distribution are parts of descriptive statistics. The hypothesis estimation and testing belong to inferential statistics. All the above, along with any other statistical tools or operations for drawing inferences or making decisions based on the data, are known as statistical analysis of data. Statistical knowledge is mainly derived from the information obtained by considering and processing a small number of representative instances and generalizing the result to all the available data. Statistical analysis of data is influenced by the data’s nature and the analysis’ ultimate goal. Some crucial aspects that affect the analysis are the type of the data (e.g., numerical, categorical, interval) and the data distribution (if the data follow a normal distribution or not), which influences the choice of the statistical tests to be used. Another factor that influences the statistical analysis is the sample size; the more limited the data sampling, the larger the expected error in the result. And lastly, the computational resources and, in some circumstances, ethical considerations [65].

Statistical Tests

Some of the available tests that are used in this thesis are the following:

1. Kolmogorov-Smirnov test

   There are several procedures for testing the normality of a population of data. Data normality testing is essential to decide which statistical test to operate. For this purpose, the Kolmogorv-Smirnov is used. The K-S is utilized to determine whether the distribution of the data sample simulates the normal distribution given a normal distribution as a reference. It is also generally used to determine whether the distributions of two different data samples are alike (look-alike) or differ from each other [65].

2. Statistical T-test (and ANOVA)

   A T-test is a procedure that compares the mean values of two sample sets and pins if there is a statistically significant difference between them. For this test to be performed, the sample datasets must be drawn from a normal population. Another crucial assumption of this test is that the null hypothesis states that the two means are equal, while the alternative is that they are not equal. Some types of t-tests are the following: The one-sample t-test is preferred when it must be clarified if the mean of a sample dataset is the same or differs from the source of truth (known) mean value of a population. Next is the paired samples t-test, which compares the two mean values of two related sample datasets (e.g., measuring a parameter of a group of patients, such as blood sugar, before and after a specific treatment). The last t-test is the independent one or the unpaired t-test that is utilized to decide if there is a statistically significant difference between the mean values of two independent sample datasets. Ultimately, there is the ANOVA test, which does what the t-test does but for two or more groups and compares the mean values of two or more sample datasets [70].

3. Wilcoxon signed-rank test and Wilcoxon rank-sum test

   These tests are used when the sample distributions are not normal or unidentified and when the data, in addition to being continuous numerically or quantitatively, specify rank order (ordinal), such as medium-good-very good, or are nominal, such as female-male.

   The primary goal of the Wilcoxon rank-sum (Mann-Whitney U) test is to compare two independent sample sets (from different populations) to define if there is a significant difference between their distributions. It is important to notice that the test does not consider a normal distribution but supposes that the two samples have similar shapes. It estimates whether the distributions of the two groups are the same or if one tends to have higher values than the other. After applying the test, the outcome is a U statistic and a p-value. Generally, a low p-value implies a significant difference between the two groups.

   On the other hand, the most crucial purpose of the Wilcoxon signed-rank test is to compare two related samples or matched pairs to determine if there is a significant difference between their distributions. For example, the related samples can occur from measuring a parameter before and after the treatment. It is often applied when the data is not normally distributed or when dealing with ordinal data. The same as before, the test does not assume a normal distribution, but it does assume symmetry in the distribution of differences between paired observations; this is the null hypothesis. Moreover, the test provides a signed rank statistic and the p-value [65].

2.5 Deep Learning

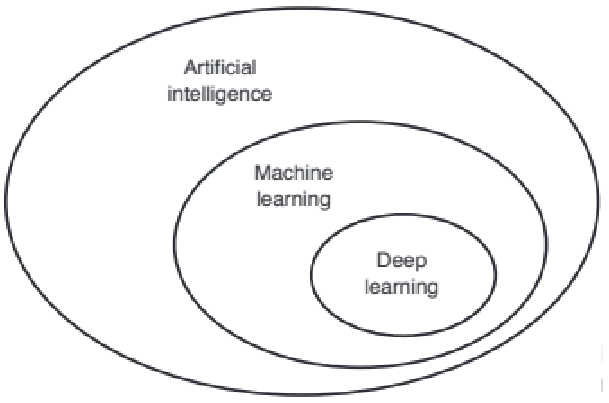


Figure 19. Machine learning and Deep learning [71]

Deep learning constitutes a subcategory of machine learning, as shown in Figure 19, that focuses on creating meaningful representations from data. Deep learning refers to training neural networks, like MLP (Multi-Layer Perceptron) that had been described above (classifiers 6, 7), but with more than two non-output (more hidden) layers [37]. This approach highlights the use of consecutive layers. The number of layers used in a network is known as the depth of the network. These neural networks often incorporate tens or hundreds of layers, all learned automatically from the training data. In contrast, machine learning algorithms (described previously in §2.4.1) typically focus on one or two layers, and they are occasionally referred to as shallow learning.

2.5.1 CNN (Convolutional Neural Network)

The feed-forward neural network, known as the convolutional neural network, can extract features from data having convolution structures. Unlike traditional methods of feature extraction, CNN can extract features automatically without the need for manual intervention. The basic CNN convolutional neural network comprises the following layers: multiple convolutional layers, pooling layers, fully connected layers, and, eventually, the final layer, which is the output. This architecture is generally used in image classification problems.

The CNN architecture's first, most basic, and most important component is the convolution layer. These layers contain/include filters (kernels) that are applied to the input images, which are 2D matrices, and implement convolutional processes. So, it convolves or multiplies the pixel matrix generated for the given image or objects to build a feature map. Storing all the unique features of an image while reducing the data to be processed is the primary benefit of a feature map. After every convolutional operation, an activation function is usually utilized. The most often activation function in CNNs is the Rectified Linear Unit (ReLU). It adds non-linearity to the network by setting negative values to zero and leaving positive values intact. This activation function enables the network to capture complicated associations between the input data and the learned features. Information is lost at the edges when choosing a convolution kernel of a particular size. This is why padding is required to increase the input with zeros and indirectly adjust the size. Additionally, to control the convolving density, stride is used. The stride distance for each convolution can be adjusted to control the amount of shift. If a larger stride is chosen, the output volume will be smaller. For instance, a filter with a stride of 2 will move 2 columns after each convolution. In order to achieve better results, smaller strides can be employed. Increasing the stride results in lower density. To avoid redundancy, an important step, pooling, is suggested. So, pooling is used to reduce the feature map's dimensions further, keeping only the essential features while reducing the spatial invariance. This is helping in fixing the problem of overfitting. One of the most used and preferred techniques is max pooling, which considers the highest value from each sub-matrix of the feature map and composes an independent matrix from it. Doing this ensures that the learnable features stay limited while preserving any image's key features. After the pooling is accomplished, one or more fully connected layers are typically used. These layers connect every neuron from the previous layer to the successive layer, allowing the network to learn patterns and make the final predictions.

This basic CNN architecture, shown in Figure 20, can be extended and modified to adjust more complicated tasks and variations in network design.

There are multiple CNN architectures available, such as LeNet, AlexNet, VGGNet (VGG-16 and VGG-19), GoogLeNet (Inception v1, v2, v3), ResNet (ResNet-18, ResNet-34, ResNet-50, etc.), and MobileNet (MobileNetV1, MobileNetV2, and MobileNetV3). These architectures vary in the number of layers they contain and the sizes of their trainable parameters [12].

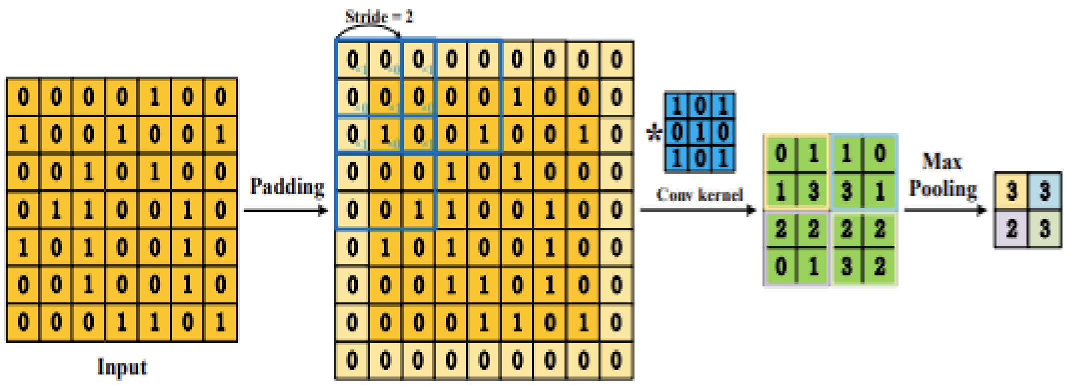


Figure 20. Illustration of the steps to build the CNN: 1. Padding, 2. Multiplying using a filter with stride, 3. Max pooling [72]

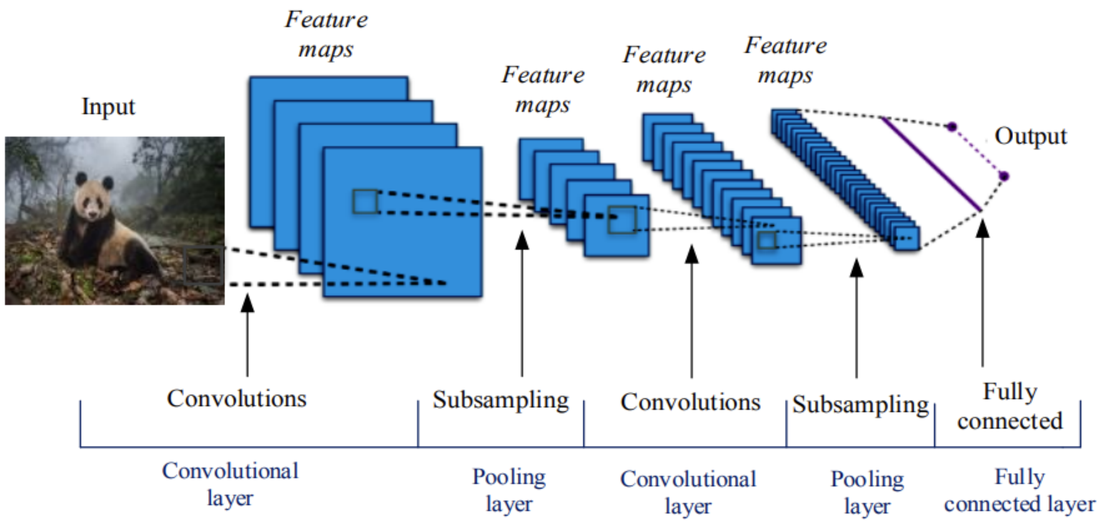


Figure 21. Visual representation of a basic CNN architecture [73]

2.5.2 VGG16

VGG16 was generated by the Visual Geometry Group of the University of Oxford in 2014 and is a convolutional neural network. It contains 16 [convolutional] layers that have some weights, which explains its name. Thus, it is defined as a deep neural network. This network has about 138 million parameters. As it is shown in Figure 22, the input layer is usually a 224x224x3 RGB image. Thirteen convolutional layers, which have a fixed size (3x3), follow the input layer. Also, VGG16 contains five pooling layers that reduce the size of the feature maps by applying the max pooling method, which takes the maximum value within a (2x2) region. The final layers are the three fully connected layers, where the first two include 4096 hidden layers and the last 1000 hidden layers. Furthermore, VGG16 uses a softmax layer, applied to the last fully connected layer, to generate a probability indicating the probable class for the input image. The structure of VGG16 is a continuation of the classical convolutional network, but it extends the network's depth by utilizing a 3×3 convolution core flexibly, resulting in improving the network's performance. Also, using smaller convolution cores can decrease the number of network parameters [74]. However, the VGG16 model also has some weaknesses, such as that the fully connected layer has numerous parameters, which occupy much memory and require considerable computing resources [75].

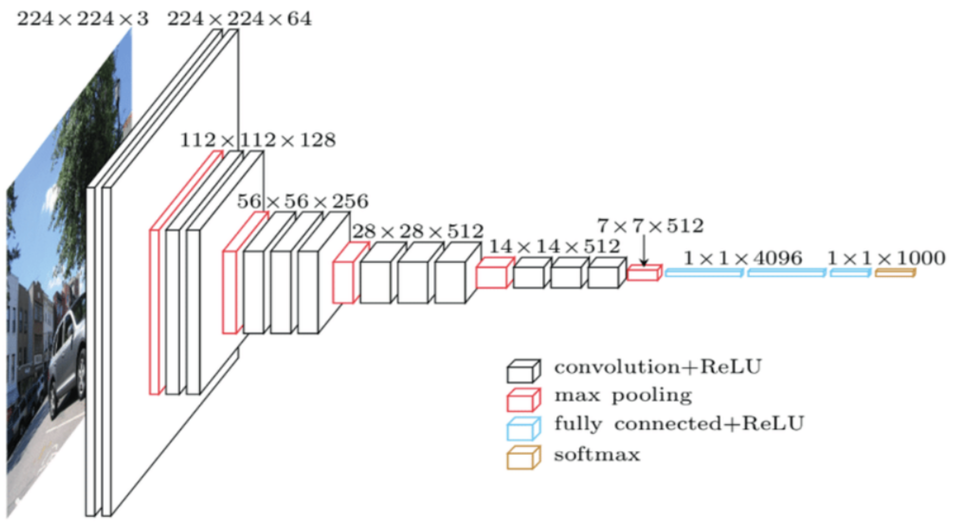


Figure 22. VGG16 architecture showing the input layer, the intermediate layers, and the final layer [75]

2.5.3 Mobilenetv2

Google has developed a range of lightweight models called MobileNets for use on embedded devices and mobile phones. These models are built using depth-wise separable convolutions and advanced techniques.

The MobileNetV2 was introduced in 2018 as an improvement over the initial iteration of the MobileNet architecture that depended on depthwise separable convolution as its foundation. The depthwise separable convolution splits the convolution process into two distinct procedures: the depthwise convolution and the pointwise convolution. With this separation, there is a notable decrease in the number of parameters and computations needs, all while keeping a high level of accuracy.

MobileNetV2 outperforms the first version of MobileNet by introducing a new concept that adds inverted residuals with a linear bottleneck design. The foremost component that constitutes the MobileNet is the input layer, which is an image, commonly of size 224x224 or 128x128. Then, there is the initial convolution. The MobileNetV2 begins with a typical 3x3 convolutional layer (with a stride of 2), which decreases the spatial dimensions of the input [40]. Furthermore, the most important blocks of MobileNetV2 are the inverted residual blocks. The number of these blocks is 19. Every one of these blocks includes a 1x1 pointwise convolution to expand the number of channels, followed by a depthwise separable convolution, and ends with another 1x1 pointwise convolution to reduce the channels to reach the initial number of channels. There are also skip connections to connect the input of the block to its output which allows the original input information to be directly connected with the output of the block. The first version, which utilizes original residual blocks, follows a wide-narrow-wide strategy, which is also shown in Figure 23 (a). In contrast, MobileNetV2 follows a narrow-wide-narrow strategy, which is also described as a bottleneck design, as shown in Figure 23 (b). This means that the block's intermediate layers have fewer channels than the input and output layers.

MobileNetV2's last component involves a global average pooling layer, trailed by a fully connected layer that utilizes softmax activation to perform classification.

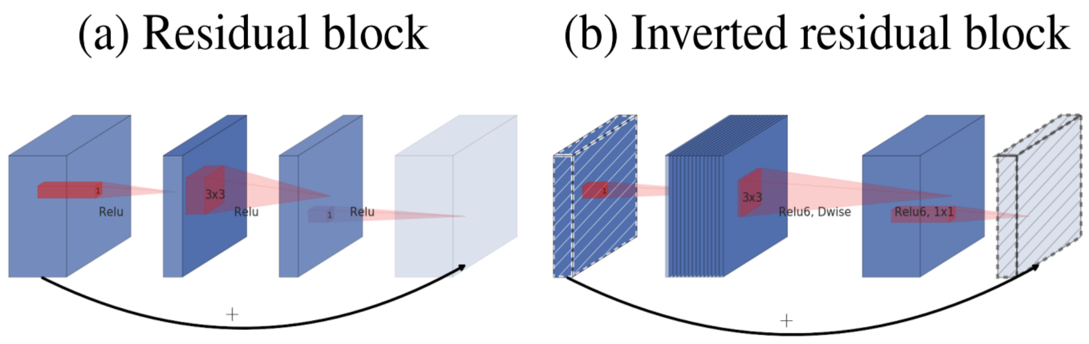


Figure 23. Core blocks of MNV2 architecture: (a) Residual block and (b) Inverted residual block [76]

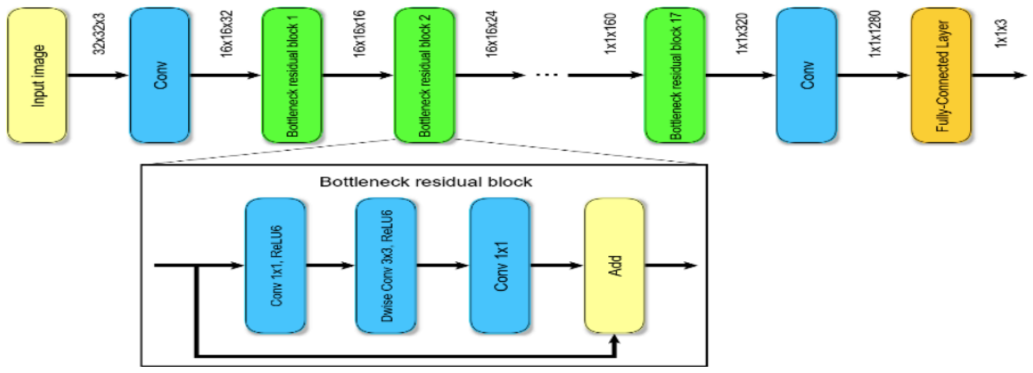


Figure 24. Graphical description of an MNV2 architecture [12]

Last but not least, MobileNetV2 introduces a width multiplier parameter (w) that allows managing the number of channels in each layer. This parameter w is found between this range (0, 1] and has characteristic settings of 1, 0.75, 0.5, and 0.25, where w = 1 is the baseline MobileNet and w < 1 is the reduced MobileNets. Modifying this parameter allows the model size and computational complexity to be scaled up or down. A smaller width multiplier reduces the number of channels, resulting in a smaller and faster model [77].

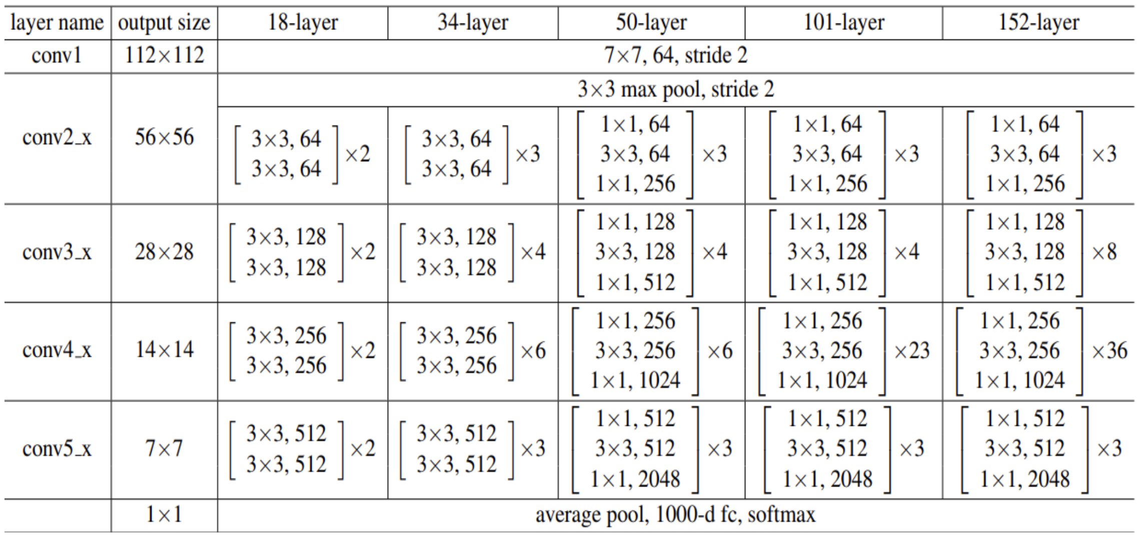
Compared to traditional CNN architectures, MobileNetV2 effectively balances accuracy and efficiency by reducing computational complexity. Hence, it is utilized in many applications, especially when there are limited computational resources or power efficiency is a top priority, such as in mobile devices, embedded systems, and real-time applications.

2.5.4 Resnet50

ResNet-50 is a famous convolutional neural network design widely used for diverse computer vision tasks like image classification, object detection, and image segmentation. A member of the ResNet group, it was created to combat the issue of degradation that arises with deep neural networks. The network's name, which includes ”50”, denotes its number of layers. Although Resnet-50 has 50 layers, it contains less than 23 million trainable parameters, which is comparatively smaller than other architectures [78].

The degradation problem refers to the phenomenon where increasing the depth of a network leads to decreasing performance or accuracy. In traditional deep networks, as more layers are added, the network can struggle to optimize and learn effectively. ResNet addresses this issue by introducing skip connections, also known as residual connections or shortcuts. The critical innovation in ResNet-50 is the residual block. A residual block consists of a series of convolutional layers followed by an identity skip connection. The skip connection allows the network to learn the difference, or the residual, between the block's input and output. By adding the residual to the output, the network can learn to focus on the changes needed to be made to the input to obtain the desired output [75].

Typically, as it is displayed in Figure 25, a ResNet50 neural network is constructed of a convolutional layer, a max pooling layer, four residual blocks, a global average pooling layer, and the fully connected layers. The 7x7 convolutional layer with stride 2 performs the initial input image processing. After that, the max pooling layer with a 3x3 kernel and stride 2 is applied to reduce the spatial dimensions. The 4 residual blocks contain multiple residual units. A group of convolutional layers is included in every residual unit, which is followed by batch normalization and ReLU activation. The skip connections, also known as residual connections, avoid the convolutional layers and combine the initial input with the block's output. Afterward, the global average pooling layer is applied to reduce the spatial dimensions to 1x1. This operation aggregates the features across the spatial dimensions while preserving the channel information. Lastly, the FC layers follow the global average pooling layer. These layers perform the final classification based on the learned features.



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				

Figure 25. Different ResNet architectures for various layers [79]

Figure 26 below shows the architecture of ResNet50. The filters' dimensions and quantity of layers may vary within each block, but the overall architecture remains the same. The structure of ResNet-50 is unique, featuring skip connections that allow the training of deep networks and enhance gradient flow. The skip connections help the vanishing gradient problem, allowing ResNet-50 to achieve high accuracy even with its depth.

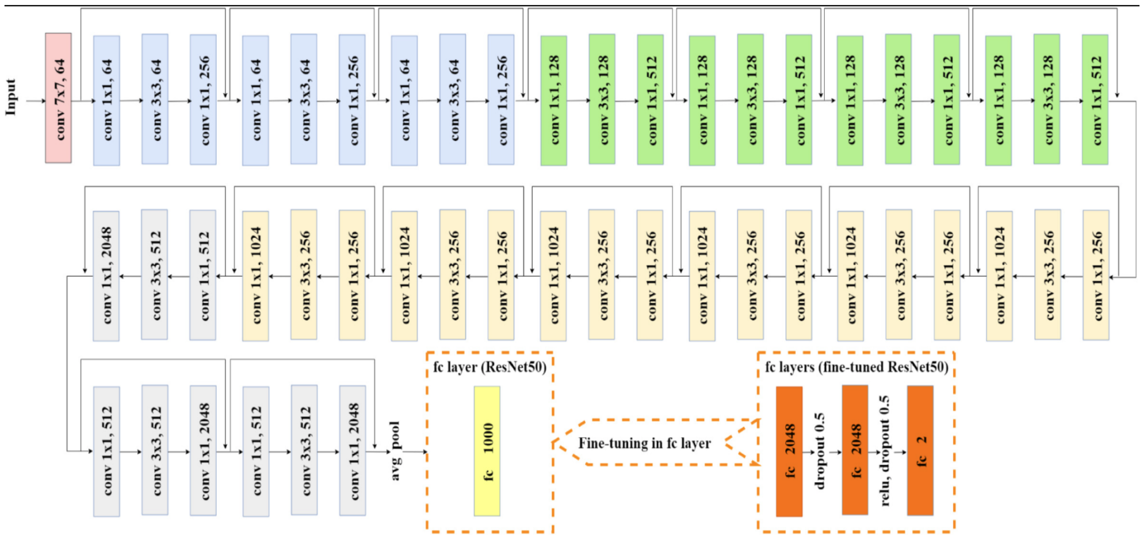


Figure 26. Graphical description of an RES50 architecture [80]

2.5.5 InceptionV3 (GoogLeNet)

In 2014, Google researchers created the Inception architecture, which is also called GoogLeNet. This CNN architecture was designed to tackle the difficulty of effectively capturing spatial information and computational efficiency in deep neural networks. Inception is based on the concept of utilizing "inception modules" that comprise numerous convolutional operations, each utilizing different filter sizes in parallel. The purpose of these parallel operations is to capture features at different spatial scales, thereby providing the network with a comprehensive set of features to learn from. Various filter sizes within a single module enable inception to capture both intricate details and broader contextual information simultaneously [81].

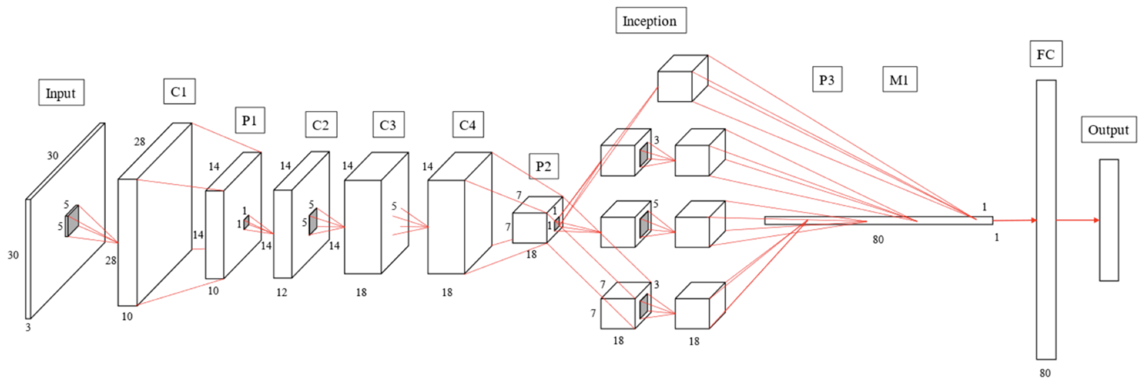


Figure 27. Visual representation of an Inception/GoogLeNet architecture [82]

The network starts with a module that performs initial preprocessing and feature extraction on the input image. This module usually consists of several convolutional and pooling layers that are utilized to capture low-level features. The main structure of InceptionV3 consists of multiple

Inception modules that are stacked one after the other. Each module has several parallel convolutional branches of different filter sizes. The network can capture features on various scales, including local details and broad contextual information with these branches. Inception modules rely on a combination of various convolutions, such as 1x1, 3x3, and 5x5, and pooling operations to extract and process features in parallel. Also, throughout the network, there are reduction modules that are usually positioned after the inception modules. These reduction modules work towards decreasing the feature maps' spatial dimensions and simultaneously expanding the number of channels. The reduction modules usually consist of pooling operations. InceptionV3 contains supplementary classifiers connected to the intermediate layers designed to enhance gradient flow and provide regularization during training. While the main classifier predicts the ultimate classes, the auxiliary classifiers play a role in preventing overfitting by contributing to the overall loss [83]. At the final stage, the network utilizes a global average pooling layer, which minimizes the spatial dimensions of the feature maps to 1x1. Following this, the pooled features are inputted (combined and fed) into fully connected (FC) layers, determining the conclusive classification established on the acquired features.

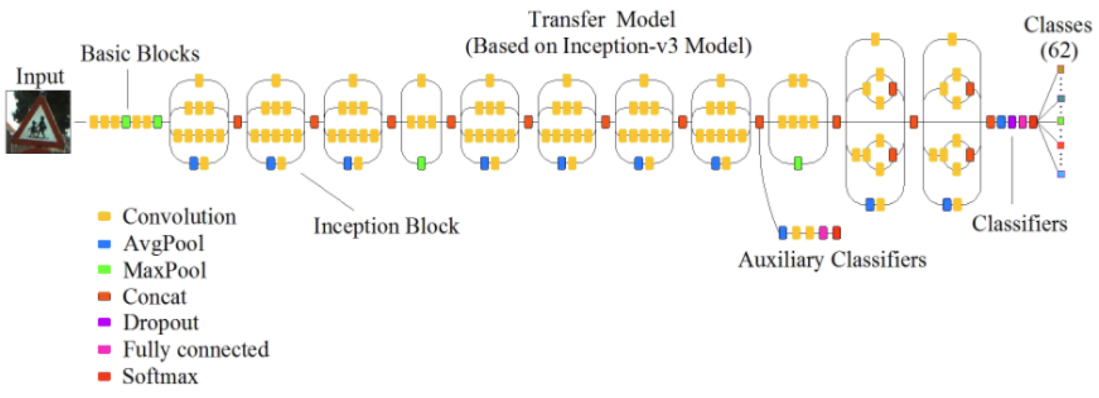


Figure 28. Detailed representation of the architecture of the INCV3 network including also the auxiliary classifiers [83]

InceptionV3 achieves high accuracy by using the power of the parallel branches in the Inception modules. It effectively captures features at different scales. The architecture's design balances depth and computational efficiency, making it a good fit for research and practical computer vision applications.

2.5.6 EfficientnetB0

The EfficientNet architecture consists of a family of convolutional neural networks (EfficientNetB0 -EfficientNetB7), among which the EfficientNetB0 serves as the base model. The objective of these networks is to provide high-quality results while utilizing minimal computing resources. Among all the variants, EfficientNetB0 is the most economical in terms of computational resources required for its implementation. EfficientNet is characterized by compound scaling, which involves balancing and principled scaling of the network's depth, width, and resolution. In line with this approach, EfficientNetB0 has been optimized to ensure optimal performance and efficiency.

The architecture of EfficientNetB0 can be seen in Figure 29 and begins with the input image, typically of size 224x224 or 299x299. The first layer contains a convolutional layer that processes the input image and extracts basic features. The convolution layer follows the inverted residual blocks that were introduced in §2.5.3. Every block consists of a combination of depthwise separable convolutions and pointwise convolutions. These operations allow the network to notice local and global features while lowering computational costs. EfficientNetB0 integrates compound scaling to optimize its size and performance. The network's depth, width, and resolution can be scaled simultaneously. The depth increases by adding more inverted residual blocks, the width increases by multiplying the number of channels in each layer, and the resolution is kept at its default value. After the series of inverted residual blocks, EfficientNetB0 employs a global average pooling layer, which lessens the spatial dimensions to 1x1 while maintaining the depth information. This pooling operation allows the network to aggregate information from the entire feature map. Lastly, EfficientNetB0 has an FC layer with softmax activation to accomplish the classification, which is also the final layer.

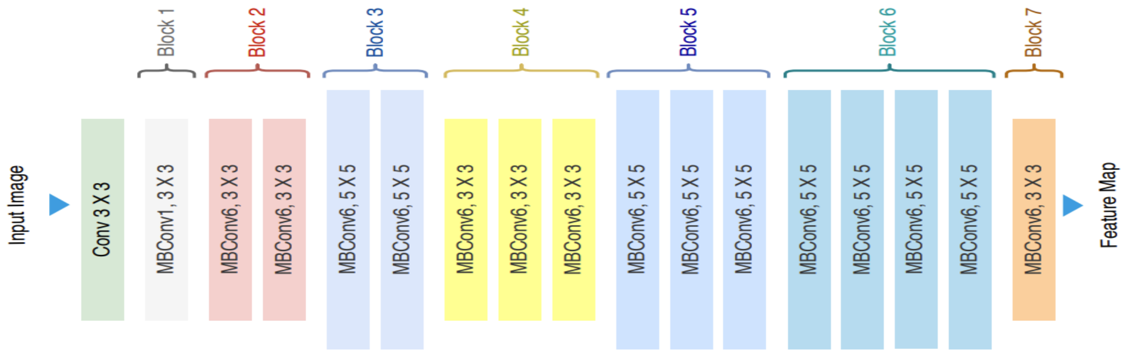


Figure 29. Diagram illustrating of EfficientnetB0 architecture [84]

Every following architecture is similar to its premature version. The diverse feature maps are their single distinction as they grow the number of parameters [84]. EfficientNetB0 serves as the baseline model of the EfficientNet architecture, providing symmetry among the model size and the computational efficiency. Also, it demonstrates competitive performance on image classification tasks, requiring fewer parameters and computations/calculations than other models.

2.6 Related efforts

Other researchers have addressed the topic or related topics to the present thesis in the past. Thus, other similar studies have been noted with varying results, some of which are summarized in this chapter. The results of these studies have helped the scientific community make discoveries about the condition of CRC. Such research aims to gain an understanding of the condition and find new ways to contribute to a more straightforward diagnosis and create comprehensive treatment plans.

The same dataset used for the present thesis's purposes was also used in 2016 for "Gland Segmentation in Colon Histology Images: The GlaS Challenge Contest." This paper presents how different techniques and algorithms address a segmentation problem. So, the paper describes how medical image computing specialists came together for the Gland Segmentation in Colon Histology Images (GlaS) challenge. The goal stood to examine the issue of gland segmentation in digital images of tissue slides dyed with Hematoxylin and Eosin (H&E). Medical image computing specialists developed algorithms to segment glands in benign tissue and colonic malignancies. The dataset used for the training process and ground truth annotations from an expert pathologist were provided, which the participants used to optimize their algorithms. The algorithms were then tested and judged based on how closely they matched the pathologist's manual segmentation. At this challenge, nineteen admissions were proposed, and the paper presents ten leading entries (the best ten models) after evaluating them [85].

In the study by Sarwinda et al. (2021) [86], which also used the same dataset as we use, the authors explore the application of deep learning methods, explicitly employing ResNet-18 and ResNet-50 architectures for detecting CRC through image classification. Focusing on colon gland images, the models are trained to differentiate the benign from malignant category. Also, three different testing datasets are utilized to evaluate the networks. The results demonstrate that ResNet-50 consistently exceeds ResNet-18 in the case of the evaluation metrics across all available testing datasets. The classification accuracy exceeded 80%, the sensitivity 87%, and the specificity 83%. The study concludes that the deep learning approach, particularly utilizing ResNet variants, generates highly reliable and reproducible outcomes for analyzing biomedical images of colorectal cancer. The preprocessing stage also involves the Contrast-Limited Adaptive Histogram Equalization (CLAHE) application to improve the image's features by addressing low contrast in grayscale images.

Another proposed approach, explained by Ponzio et al. [87] in 2018, explores the challenge of colorectal cancer (CRC) diagnosis via histological image analysis. The authors try to enhance the accuracy and efficiency of classification, considering the complexity of histological images. This paper uses a dataset of H&E-stained whole-slide images of colorectal tissues containing healthy tissues, adenocarcinomas, and tubulovillous adenomas. Moreover, it uses a CNN for automatic classification, specifically the VGG16 architecture. Then, the CNN is fully trained on CRC samples and investigates transfer-learning approaches utilizing the VGG16 (pre-trained model). The main focus is distinguishing between healthy tissue, adenocarcinoma, and tubulovillous adenoma. The key findings are that it achieves around 90% classification accuracy when the CNN is fully trained on the CRC dataset. Also, the transfer learning approaches, especially fine-tuning the VGG16 after POOL3, outperform full training, reaching an accuracy of about 96% on the test dataset. Finally, full training is computationally intensive, while transfer learning provides better results with significantly less training time. The study highlights the potential of DL, particularly transfer learning, in enhancing the accuracy of CRC histological image classification. The findings suggest that pre-trained models can effectively extract features for CRC diagnosis, offering a more efficient alternative to full training.

The paper published by Naresh Kumar et al. [88] notes the critical issue of cancer-related deaths worldwide and underlines explicitly the need for research in biomedical health informatics for early lung and colon cancer detection. In this study it is being compared the analysis of two different feature generation methodologies, the handcrafted features and the deep features extracted using CNNs. The color, texture, shape, and structure were the bases for creating six procedures that introduced the handcrafted features, employing classifiers such as Gradient Boosting, SVM-RBF, Multilayer Perceptron, and Random Forest to resolve the problem of classification of lung and colon cancer. Antithetically, the deep learning approach uses the transfer learning method with the seven deep learning frameworks for feature generation from histopathological images. Viewing the results indicates that the classifiers' performance with the deep features generated by the networks is improved compared to handcrafted features. Especially the Random Forest classifier with features generated from the DenseNet-121 showed outstanding evaluation metrics, reaching 98.60% accuracy and recall, 98.63% precision, 0.985 F1 score, and 0.1 ROC-AUC. It is concluded that the spreading of color and the color (itself) in cancer cell analysis are the most critical parameters, particularly in feature extraction. Also, the usefulness of transfer learning is underlined, allowing for improved disease diagnosis with reduced expertise, effort, and cost. Finally, a limitation considered is the lack of stain normalization, and it is suggested for future work, as well as the examination of dimensionality reduction techniques for the color images.

Chapter 3

Material and Methodology

3.1 Tools and software

For the present thesis, the programming part was held on a desktop computer with a Linux operating system, particularly Ubuntu 20.04.3 LTS version, with available 32 GB Random Access Memory (RAM) and processor (CPU) Intel Core i7-7700 @ 3.60GHz.

Foremost, **Python**, a programming language, had been chosen to implement all the necessary coding. Python is one of the most trending programming languages for scientific computing, data science, and machine learning because it owns some essential characteristics. First of all, python is fast and easy to learn and understand. It includes many useful open-source libraries that are perfect for data science and, in general, for data manipulation. Some of them that have been used in this thesis are NumPy, pandas, and scikit-learn [89].

Python programming is easier with **PyCharm**, an integrated development environment (IDE). PyCharm offers a graphical debugger, an integrated unit tester, code analysis, and version control system integration. It also supports web development using the Django framework. It is a platform-independent software functioning on different operating systems like Microsoft Windows, macOS, and Linux. PyCharm has a Professional Edition and a Community Edition. The PyCharm Professional Edition, used to develop the necessary code for the present thesis, offers more features than the Community Edition.

The most important libraries that Python possesses and which have been used are the following:

1. **OpenCV-Python**

   OpenCV is a community-produced extension (-library) that supports various programming languages (including Python, C++, Java, etc.), is available, and can be installed on different operating systems (Windows, Linux, Android, iOS, etc). Currently, OpenCV contains multiple algorithms that are expanding daily, correlated to computer vision and machine learning. In addition, it allows us to comprehend the method by which images and videos are saved, as well as how we can alter and obtain data from them [90].

2. **NumPy**

   NUMeric PYthon is the essential package for scientific computing in Python. This library offers an array object that can handle multiple dimensions, along with other objects like matrices and masked arrays. Additionally, it provides a range of functions for fast and efficient manipulation of arrays, including sorting, I/O, selecting, discrete Fourier transforms, mathematical and logical operations, basic statistical operations, elementary linear algebra, random simulation, and more. It is gratis, as it is an open-source library. This tool surpasses Python lists regarding speed and efficiency for these operations. The main advantage of utilizing NumPy is that it requires less memory and storage space. Additionally, the performance in terms of execution speed is superior. Despite this, it remains user-friendly and convenient. Furthermore, connecting existing C code to the Python interpreter is hassle-free.

3. **Pandas**

   It is used when operating with labeled or relational data, and it is an open-source library. It offers a range of user-friendly operations and data structures to manipulate numerical data and time series. It was created as an extension of the NumPy library (which constitutes a dependency) and boasts speed and increased performance, aiming to increase user productivity. This open source has numerous advantages, including the following: It efficiently manipulates and analyzes data quickly. It loads data from various file sources into a tabular data structured format with labeled axes (rows and columns) which is called DataFrame. It can effortlessly handle the absence of data, whether in floating point or non-floating point data, expressed as NaN (Not-a-Number). DataFrames can be modified in size by inserting or deleting columns and rows. They also allow merges, joins, reshapes, and shifts of datasets according to the corresponding needs. Lastly, perform split-apply-combine operations on datasets with powerful group-by functionality.

4. **SciPy**

   Python has a library called SciPy that offers essential building blocks for modeling and tackling scientific problems. SciPy features various algorithms for solving multiple problems, such as eigenvalue problems, differential equations, algebraic equations, interpolation, and others, and also enhances optimization (for the procedures). Additionally, it provides specialized data structures like k-dimensional trees and sparse matrices. SciPy is made on top of NumPy, forming the base for higher-level scientific libraries such as scikit-learn. Scientists, engineers, and people worldwide rely on SciPy [91].

   When it comes to mathematical and numerical analysis, both Numpy and SciPy are commonly used. Numpy is particularly useful for fundamental operations like sorting and indexing due to its array data, while SciPy encompasses all numeric data. In terms of functionality, Numpy provides several functions for resolving linear algebra and Fourier transforms. At the same time, the SciPy library offers a comprehensive version of the linear algebra module along with multiple other numerical algorithms.

5. **Scikit-learn (Sklearn)**

   The library Sklearn is highly effective and reliable for performing machine learning tasks in Python. Its interface in Python offers a consistent approach to statistical modeling and efficient tools for performing diverse tasks such as regression, classification, clustering, and dimensionality reduction. Sklearn depends on NumPy and SciPy packages. An essential advantage is that users with various backgrounds, even those without statistics or computer science knowledge, can easily access it. The software and web industries and fields such as biology and physics can benefit from Sklearn's user-friendly interface and advanced data analysis capabilities [92]. There are various reasons why Scikit-learn stands out from other machine-learning toolboxes used in Python. The most important is that it incorporates compiled code for efficiency. Also, it focuses on imperative programming (imperative programming suggests how the program should achieve the result).

6. **Matplotlib**

   Python's Matplotlib package is ideal for visualizing data. It is community-produced, and it is very customizable and extensible. Also, it provides a variety of plots, such as line, scattered, bar, and radial plots, with a high degree of customization and annotation. Using the versatile artist module, developers can create any visualization they desire. It permits both interactive and non-interactive plotting and provides the option to save images in various output formats such as PNG (portable network graphics) and more. For regular use, Matplotlib has a simple object-oriented interface called the pyplot module for easy plotting. Additionally, Matplotlib supports an interactive interface, which is useful for creating web-based applications and a wide range of plots. Therefore, it can operate in non-graphical environments, such as scripts, be integrated into graphical software applications, and be utilized on websites [93].

7. **Tensorflow**

   It is a publicly available library for quick numerical computing built and maintained by Google. It is ideal for development, research, and production systems. TensorFlow can operate on single CPUs, mobile devices, GPUs, and distributed systems with hundreds of machines. Thus, TensorFlow is another powerful numerical library, and it is possible to create deep learning models directly in TensorFlow. Also, a wrapper library (e.g., Keras) can simplify TensorFlow's lower-level details and complexity [94].

8. **Keras**

   Keras is a Python library for DL intents, which operates on top of TensorFlow and is designed for research and development purposes. It aims to simplify and expedite the evolution of deep learning models. Additionally, it can execute flawlessly on both CPUs and GPUs, provided that the underlying frameworks are present. Keras wraps the TensorFlow library, abstracting its abilities and obscuring its complexity. Moreover, the minimalistic way Keras is created permits the rapid definition of DL models. A Keras DL model can be designed, compiled, fitted, and then evaluated or utilized to generate predictions. [94].

3.2 Dataset

The dataset chosen and downloaded is the Warwick-QU dataset, which was publicized by Warwick University for GlaS@MICCAI'2015 [95]: Gland Segmentation Challenge Contest. Sirinukunwattana and his team in [39] designed the Random Polygons Model (RPM) algorithm, a segmentation algorithm, to extract glandular structures from colon tissue histology images, and this was the first publication employing this particular dataset. Nevertheless, this thesis will utilize this dataset to address classification problems. When downloading the dataset, an Excel sheet is also included that indicates which images belong to the benign category and which to the malignant category. Moreover, this Excel sheet has an extra column that shows that the images can be further categorized into five categories (healthy, benign/adenomatous, moderately differentiated (grade I), moderately to poorly differentiated (grade II), and poorly differentiated (grade III)). Thus, this dataset can classify the images into two or five categories. The dataset consists of 165 images in total and samples for the benign and malignant categories can be seen in Figure 30. From the entire dataset, the benign category consists of 74 images, while the malignant category consists of 91 images. Every image has a mask that annotates the gland's area, as shown in Figure 31 below.

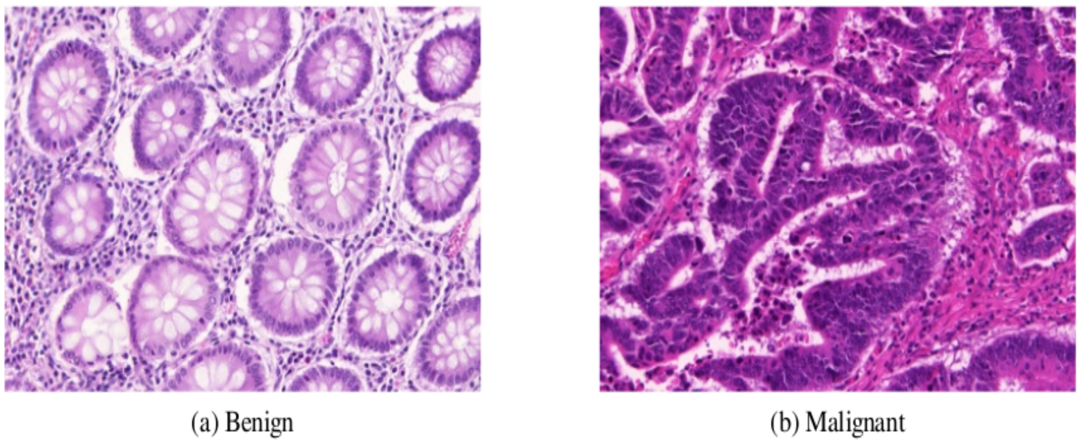


Figure 30. Histopathological images from our dataset: (a) benign and (b) malignant [39]

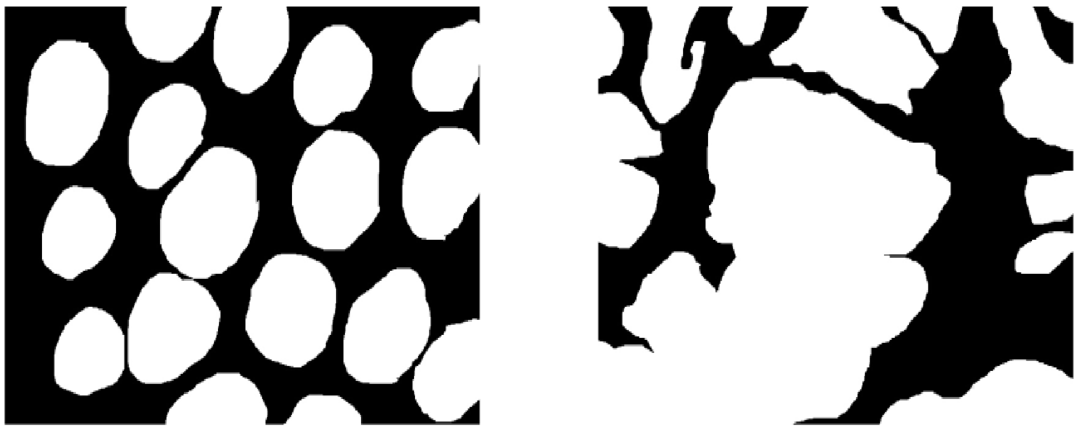
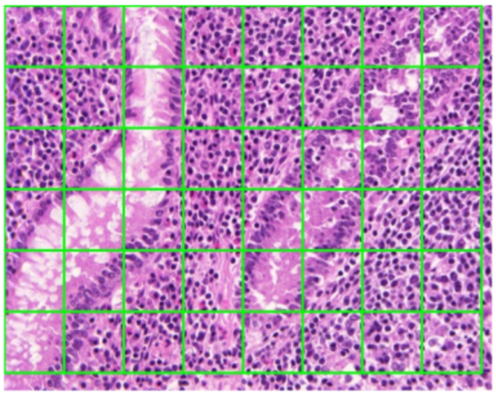


Figure 31. The mask samples that display where exactly are located the lesions and malignancies [39]

For the statistical analysis (extracting the features) and the creation of the classification machine learning procedure, we wanted to expand the number of images. Thus, we cropped the images into patches. George Xenogiannopoulos implemented the cropping algorithm for his Ph.D. study. The algorithm can accept images with their annotation masks as input and give the patches with the regions of interest (ROIs) as output. The ROIs are the areas of significant importance for the research and analysis. Also, it is important to mention that every patch takes the labels of its parent image. Therefore, considering the masks, we took the patches from this dataset from the areas containing the lesions and the malignancies. Using the gland annotations masks, we discarded the patches with less than 70% mask coverage (as shown in the following Figure 32), and we ended up with the final number of patches for our study.

After experimenting, we chose a 78x78 patch size, which is the minimum input size for most neural networks.


(a) Patches without mask

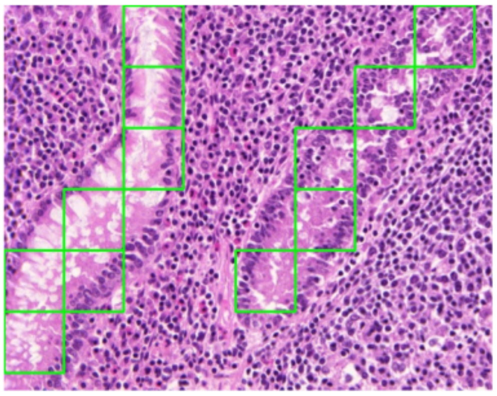

(b) Patches with mask

Figure 32. Patches images:(a) without applying the masks and (b) by applying the masks [39]

3.3 Implementation

3.3.1 Statistical Analysis - Feature Extraction

After gathering all the patches, the first step in our implementation was the careful feature generation from these patches. Focusing on our patches (1303 benign and 1891 malignant) from the merged categories, we initiated the procedure of extracting the features outlined in the theory and successfully counting 68 distinct features. After uploading and reading the images, using the OpenCV library from the folders stored in our computer, we converted the colored images into gray-colored images. This conversion of colored images into gray-colored is a standard and widely accepted practice in image processing for various reasons. First of all, it is essential for computational reasons, as it can simplify the feature extraction process. Grayscale images have smaller file sizes than colored ones, as they have a single intensity channel, and the latter ones hold more information considering multiple color channels (RGB). Secondly, color conversion is necessary because as it decreases the number of colors, the information is merged into one intensity channel, which can minimize the noise that might be present in each color channel. Hence, it might help reduce the noise in the images. Also, we needed grayscale images to generate the second-order features (as mentioned in section 2.3). In addition, using grayscale images might improve the generalization as more fundamental features and patterns might be discovered. Keep in mind that the ultimate goal is to find the features with statistically significant differences among the different classes (benign/malignant) that will help us differentiate them. Thus, after converting the patches with the region of interest (ROIs) into grayscale, we started to generate the features.

For every patch, we calculated the first-order statistics, four in total, using the Scipy library of Python or the functions of Pandas' library. After calculating these features, we saved these values in the feature sheet (with columns as many as the features and rows as many as the patches). Moving on to the second-order statistics, firstly using the co-occurrence matrices, we calculated the values for the following features: contrast, dissimilarity, homogeneity, ASM, energy, and correlation. As stated in §2.3.2, more features might occur from the co-occurrence matrices, but we focused on the ones mentioned above. Since from every image occur four co-occurrence matrices (0°, 45°, 90°, 135°), and thus four values are calculated for each feature, which means four (4) values x six (6) features = twenty-four (24) features for every image. However, we do not use all these features, but according to the literature, we can compute the mean value and the range value so that every feature will be represented by two values: the mean and the range. Hence, we end up with twelve features (six mean and six range values). We used part of the code developed in [96] to calculate the values of the mentioned features. We computed all four values for every feature, then we calculated the mean and range for every feature (*contrast_mean*, *contrast_range*, *dissimilarity_mean*, *dissimilarity_range*, etc.) and finally stored them in the feature sheet mentioned above.

Secondly, using the run-length matrices (four for every image, a matrix for every direction 0°, 45°, 90°, and 135°) that are calculated from the patches, we computed the following features: the short-run emphasis or SRE, the long-run emphasis or LRE, the Grey-level non-uniformity or GLNU, the run-length non-uniformity or RLNU, and the run percentage or RP. Using part of the code found in this source [96], we calculated the mean value and the range for every one of the five features mentioned above from the run-length matrices exactly done like in co-occurrence matrices. Therefore, in the end, we have ten features from the R-L matrices and save them in the feature sheet. So far, we have calculated 26 textural features (the 1st and 2nd order characteristics). Another group of features to add to our set of features are the LBP features (ten in number), which we calculated using the *skimage.featurelocal_binary_pattern* Python's package. The parameters that we use, as mentioned in [97], are:

- the image=our patch (a 2-D grayscale image),
- The P=8 (Number of circularly symmetric neighbor set points),
- The R=1 (Radius of the circle (spatial resolution of the operator),
- The method is "uniform" (Uniform pattern which is grayscale invariant and rotation invariant)

So, this package calculates the LBP image, and after that, we use it to form the histogram from which ten features will occur, as explained in the theory in §2.3.3. We have also added these ten features to the feature sheet. Next, we estimated the Tamura features, four in total: coarseness, contrast, directionality, and roughness. We developed our code based on the following source [98]; we calculated the features and added their values to the feature sheet. Now, we have a total of 40 features.

We are continuing with the calculation of the wavelet features. The wavelet transformation (WT) uses specific low-frequency signals and, utilizing the convolution of these signals, encounters information about them. We utilized the module dwt2 from Python's package pywt (PyWavelets - Wavelet Transformation in Python) [99]. This module creates four images (or four matrices):

- the LL(Low-Low pass filtering of the rows/columns)
- the LH (low-pass filtering of the rows and high-pass filtering of the columns)
- the HL (high-pass filtering of the rows and low-pass filtering of the columns)
- the HH (high-pass filtering of the rows and high-pass filtering of the columns)

The features of each of the following occurring images (LH, HL, HH) are the mean, median, mode, max, min, range, median absolute deviation, and mean absolute deviation. Thus, we end up with 24 features (eight from each wavelet occurring image) based on the dwt2. Lastly, we estimated, using part of the code found in [100], the Gabor features that occur from the processing of the images using Gabor filters and considering different angles and wavelengths (frequencies). Two features are formed at each angle/frequency: the energy and the amplitude. Therefore, we took the mean value and the range of a list with five different angles and a list with five different frequencies. So, four more features have been added to our list: the Gabor energy mean, the Gabor energy range, the Gabor amplitude mean, and the Gabor amplitude range. We conclude with a total of 68 features, from which we will select those that are meaningful and helpful for our research. Also, we must point out that the final feature sheet has 68 columns (as many as the features) and rows as many as the available patches.

3.3.2 Statistical Tests - Feature Reduction

After that, we designed the supervised machine learning system. This design consists of the following steps:

1. Data normalization.
2. Feature reduction (using statistical tests and other feature reduction methods)
3. Then, we choose the best/most suited features from the reduced group of features and use them (using the feature reduction method).
4. We designed the machine learning system (select classifier and evaluation method) and evaluated its accuracy to find the best feature combination.

Normalization of the data is necessary because the various features that compose the data have values whose sizes differ a lot between the features. Thus, one characteristic, such as the mean value of the pixel intensity values of an image segment, may have values ranging between 0-255, and a second characteristic, such as the standard deviation of the pixel intensity values, may range between 0-7. Considering that the system is designed from a combination of several features, it is understood that the values of the features must be normalized to participate and/or influence the decision process of the system equally.

The features that will be used in our machine-learning system must have as poor (little) correlation between the classes as possible. Having a good discriminating ability between the classes is reasonable but optional. The statistical tests mentioned in §2.4.4 can evaluate the discriminating ability. Firstly, we performed the **Kolmogorov-Smirnov** for every one of these features for the two categories (-benign and malignant) to check which features have a normal distribution (basically, if the values of the features follow a normal distribution). Then, we perform different statistical tests based on whether the corresponding feature values follow a normal distribution. For the features that follow a normal distribution in both classes, we chose the **T-test** and the **MW-Wilcoxon test** for those not following a normal distribution in one or both classes. After that, from all the available features, we selected the features that presented statistical significance differences. These features are the ones that would be used for our machine learning system; in other words, they are the input for our machine learning system.

For the KS (Kolmogorov-Smirnov) test, we used the kstest method from the scipy.stats module. As we want to test if the sample dataset (every feature) follows the normal distribution, we assume this is the null hypothesis - the feature has a normal distribution. The level of significance we chose is 0.05, which means that if the p-value is below this threshold, we will reject the null hypothesis. In this case, the alternative hypothesis indicates that the samples of the particular feature (for the two classes) do not have a normal distribution. We performed this test for every feature for both classes and if the samples of the feature in both classes had a normal distribution, we performed the T-test. Otherwise, we proceeded with the MW-Wilcoxon test.

For both tests, the T-test and the Wilcoxon, the null hypothesis considers no difference between the two classes (benign and malignant), stating that the distributions for these two classes for a specific feature are the same. Also, the level of significance is set to 0.05, and a p-value under this point means that the specific feature shows the existence of a statistically significant difference between the classes. Then, we separate and save into another feature sheet the features that have a discriminating ability between the two classes from those that do not. At this point, we conclude with fewer features of the total 68 features we started, and this number ranges depending on which classes we are examining.

3.3.3 Investigation before working with the actual dataset

After completing the statistical analysis part, we kept only the essential features that will be used in our ML system. Thus, that phase was like a ”preprocessing” phase for our ML system. To build the ML system, we basically will use these features to train ML classifiers, then test them and see how well they were actually trained (how well they learned). The classifier must be trained with different data from those used in the testing phase. After the training, we aim to select the algorithm that best separates the benign from the malignant samples in the test data, which is the ultimate goal. Also, we need to mention that before the training phase, we will use one more feature reduction method to reduce the input features further. Considering that we have nine different classifiers, two reduction methods, and four evaluation methods, we are getting 72 possible combinations (9*2*4=72) to find which one performs the best.

Initially, we did a little investigation. The initial dataset (with the original images) can also be divided into five categories, as mentioned in §3.2: healthy, adenomatous/benign, grade I, grade II, and grade III. To choose among the reduction and evaluation methods available, we took a small sample of our dataset, 300 patches from the healthy category and 300 from the Grade III category, which we expected to be the most distinguishable. Essentially, we expect good results (acceptable accuracy).

We first normalized our data and then generated and stored all the features. Then, we performed our statistical analysis procedure to find the features with significant statistical differences and consider only them for our ML system. To encounter these features, we applied the significant test ranking procedure (regarded in theory in §2.4.2 as a feature reduction method), including applying the Kolmogorov-Smirnov test to see if every feature follows a normal distribution. We then applied the corresponding T-test (for those that followed the normal distribution) or MW-Wilcoxon, as mentioned above in §3.3.2. We have set the p-value to 0.05, and if the calculated returned p-value from the tests was less than the p-value (threshold) we set, we considered that the particular feature could help us distinguish the two classes. From this procedure, we ended up with 34 features from the initial 68.

Thereon, apart from the significance test ranking, which has already been applied, we also used the other methods mentioned in the theory in §2.4.2 for further feature reduction, and these were the correlation ranking, the mixed criterion (which combines two different reduction methods), the PCA method, and RFE wrapper method. We mainly used the PCA and RFE methods with different combinations of classifiers and evaluation methods. A reduced number of prioritized new PCA features are obtained using the PCA method. Thereafter, we built the RFE wrapper method using the RFE algorithm combined with statistical regression methods, such as RandomForestRegression, LogisticRegression, or DecisionTreeRegression, for optimal feature selection. We used from the sklearn.decomposition module, the PCA method, for our PCA algorithm. Lastly, we imported the RFE method from the *sklearn.feature_selection* module for our *RFE_wrapper* algorithm based on the RFE method and the LogisticRegression function needed from the *sklearn.linear_model*.

A subset with the best combination of features can be selected from these ranked features. Thus, we conclude with 17 total (features) using the RFE, from which we selected the first 15 to make our investigation and tests faster. We did the same thing with the PCA method; from the 34 new features calculated (by the PCA algorithm), we selected the first 15 for the investigation.

The extensive list of classifiers, like it is mentioned in theory in §2.4.1 contains the MDC, the KNN, the Naive Bayers, the LDA, the logistic regression algorithm, the MLP, the linear SVM, the CART, and the RF algorithm. We imported from sklearn.neighbors import NearestCentroid and the KNeighborsClassifier methods, from sklearn.naive_bayes the GaussianNB method, from sklearn. discriminant_analysis the LinearDiscriminantAnalysis, from sklearn. linear_model the LogisticRegression and the Perceptron, from sklearn.svm the LinearSVC, from sklearn.ensemble the RandomForestClassifier method, and from sklearn.tree the DecisionTreeClassifier method.

We tested every classifier (MDC, KNN, etc.) for up to three feature combinations, using the PCA and RFE methods as reduction methods for each of the following evaluation methods: LOO, Bootstrap, Hold-out, and K-fold. We choose to run our test for up to three feature combinations to keep computational time low. Thus, we created the following Table 4, which shows the accuracy (%) of every test we did with every classifier. As shown in Table 4, the RFE reduction method performed much better than the PCA, so we selected the RFE to continue with our original dataset. As for the evaluation method, we also can notice in Table 4 that the LOO, the K-Fold, and the Bootstrap performed the best and almost the same (they have similar mean accuracies for RFE), with LOO taking the first place among the three. Taking into consideration that regarding the LOO evaluation method, the procedure took more time, as it can be more computationally intensive, and noticing that the Random Forest classifier using the bootstrap method accomplished the best accuracy (90.63%) across all classifiers and feature reduction methods we decided to use the bootstrap method as an evaluation method for the rest of our research. Along with the bootstrap, we also decided to consider the K-fold (with the K=10) evaluation method because it uses different data for training and testing. The process is repeated k times (ten in our case) for each test set (so 10 * 10 = 100 total number of trains), and the classifier's performance is evaluated by whether it accurately classified the test data.

The whole procedure was executed five times, and the final accuracy score occurred from the mean value of those five runs.

3.3.4 Two categories

After finishing with this investigation, we moved on to our actual data, the merged categories - benign and malignant. We merged the healthy and benign/adenomatous categories into one named benign, and grades I, II, and III were merged into a class called malignant. Thus, we ended up with 1303 patches that belong to the benign category and 1891 patches for the malignant class. We normalize the data, calculate all 68 features, and store their values in two feature sheets. These sheets were the input for our statistical analysis procedure and also for the ML procedure.

For the statistical analysis, we followed the exact procedure as we did for the smaller dataset in the investigation phase. We applied the Kolmogorov-Smirnov test to every feature to see if it followed a normal distribution. Then, we applied the corresponding T-test (for those that followed the normal distribution) or MW-Wilcoxon. By the end of this procedure, we conclude with 44 features from the total 68, for whom we also created the boxplots (44 boxplots in total). We decided to show the boxplots of the most frequent features the classifiers have specified. So, by using the RFE wrapper reduction method (along with the logistic regressor) from the 44 features, we concluded with the following 22 features: [mean, skewness, kurtosis, con mean, energy range, correlation mean, correlation range, LBP6, LBP7, LBP8, RP mean, dwt2 max H1, dwt2 min H1, dwt2 range H1, dwt2 meanAbsDev H1, dwt2 range V1, dwt2 meanAbsDev V1, Tmr coarseness, Gabor energy mean, Gabor energy range, Gabor amplitude mean, Gabor amplitude range].

Every test we did for every classifier was for combinations of up to 5 features. Our system checked first the combination of 2, 3, and 4 features, but the best results were seen for a combination of 5. We did not check further for a higher number than 5 of a combination of features because the procedure was already computationally intensive, and we wanted to avoid forcing it further. Apart from that, it would also increase the complexity of drawing conclusions and explaining how the features correlate.

The classifiers were evaluated during the training of our ML system using the bootstrap and the K-fold as evaluation methods. Also, the 22 features were combined in combinations of up to 5, and the results, the mean accuracy, specificity, and sensitivity of five runs, can be noticed in Table 5. Based on the accuracy values, the classifier that best separated the data was the Random Forest, followed by the Cart and KNN classifiers. In Figures 33 and 34, the feature frequency histograms show which features appeared in the best results of all nine classifiers using the bootstrap evaluation method. The five top features were the skewness, the LBP6, the Gabor energy range, the energy range, and the correlation mean. We also created the boxplots for these features, as the following figures are displayed in section 4.1.

By completing this phase, we encounter some promising (and satisfying) results. These results for these two categories made us consider continuing our research. We continued in two directions: the first was to build a deep learning system using some neural networks (for the benign/malignant categories), and the second was to split the data into five categories.

Deep learning for two categories

We continue with our deep learning system for the two categories. Our input was the two folders containing the patches for every class (benign and malignant). Table 3 shows 1303 benign and 1891 malignant patches. We used the CNN network and the pre-trained Vgg16, MobileNetV2, ResNet50V2, InceptionV3, and EfficientNetB0.

Neural Networks

First of all, we imported the needed libraries (also libraries for monitoring and controlling the training -matplotlib.pyplot). Then, we loaded the data/images using the io.imread function, and we initialized the model. Also, we used from *sklearn.model_selection* the *train_test_split* method (to separate the training data from those used for testing). When the model is created, we can configure the model using the ‘model.compile()‘, train the model with ‘model.fit()‘, or use the model to make a prognosis with ‘model.predict()‘.

When implementing a CNN, many things must be handled, like the number of layers, filter size, padding type, and more. A solution for all of this is to use a pre-trained model for our image classification task. Many pre-trained models exist, like Resnets, Inception, Vgg, etc.

To build the **CNN** for our project, we used the Keras Sequential API, and with a few lines of code, we created and trained it. We had to add more layers after creating the model with the following line of code: ’models.Sequential()’. We formed the convolutional base of the network, utilizing the basic pattern of a stack of Conv2D and MaxPooling2D, which takes as input tensors of the image’s shape and gives as an output a 3-D tensor (height, width, channels). Also, as we go more in-depth into the network, we notice that width and height are shrinking. The first argument for each Conv2D layer (e.g., 16, 32, 64) controls the number of output channels. To complete the model, we have to pass/feed the last output tensor from the last layer of the convolutional base into the Dense layer(s). The last output layer is a 3-D tensor, and because the Dense layers take as input vectors that are 1-D, we have to convert (Flatten) our 3-D layer to 1-D. The final Dense layer must contain the number of our output classes (2 or 5 in our implementation). After building the model, we compile it, train it, and finally evaluate it (plot and save the accuracy/loss graph).

Moving on to the following pre-trained networks (VGG16, mobilenet, etc.), we used the transfer-learning concept. Generally, we selected this approach when the available dataset has fewer samples. The typical transfer learning workflow implemented in Keras is the following: First, we initialize the base model we want and download and use the weights into it. The first time we ran this model, Keras downloaded the weight files and stored them in a specific directory (/.keras/models), so the next time we run the model, it will automatically find the downloaded weights and use them. Secondly, we ”freeze” all the layers in the base model by setting *base_model*.trainable = False”. Then, we develop a new model built on top of the output generated by a single layer from the original base model. Lastly, we trained the new model on our dataset. Thus, we instantiate the base model for every one of the following models:

```
base_model = tf.keras.applications.MobileNetV2 (
    input_shape=shape,
    include_top=False,
    weights="imagenet",
)

base_model = tf.keras.models.Sequential()
base_model.add(tf.keras.applications.ResNet50V2(
        include_top=False, weights="imagenet," classes=n_classes
    )
)

base_model = VGG16(
    weights="imagenet",
    input_shape=shape,
    include_top=False
)

base_model = tf.keras.applications.InceptionV3(
    input_shape=shape,
    include_top=False,
    weights="imagenet",
)

base_model = efficientnet.tfkeras.EfficientNetB0(
    input_shape=shape, include_top=False, weights="imagenet"
)
```

Then we set the base_model as ”base_model.trainable = False”. After that, we create a model on top of every base_model selected, adding more layers and using the

```
tf.keras.layers.GlobalAveragePooling2D()
```

we convert the 3-D tensors (the previous layer's output) to 1-D vectors. We also add a Dense layer,

```
tf.keras.layers.Dense(classes, activation=activation)
```

selecting for activation ”sigmoid” if we have two classes or else ”softmax.” Furthermore, we use

```
 tf.losses.SparseCategoricalCrossentropy(from_logits=False)
```

to calculate the loss that we are going to use as an input in the

```
model.compile(optimizer=optimizer, loss=loss, weighted\_metrics=["accuracy"])
```

Lastly, after building the model, we compile it, train it, and evaluate its performance.

All the runs were performed with the following training parameters: epochs 20 and 100, patch size (78, 78), test size equal to 20 % of the data, and batch size equal to 128. So, we test them for two different numbers of epochs. The results, the mean accuracies of 5 runs, and their losses are presented in 4.1.2 section.

3.3.5 Five categories

Accordingly, after splitting the data into five categories, from benign and malignant to healthy, benign/adenomatous, moderately differentiated (grade I), moderately to poorly differentiated (grade II), and poorly differentiated (grade III), we compared and tested the different combinations in pairs with our ML and DL system.

We followed the exact same process that we analyzed before for the five categories in combinations of two, making this way ten combinations of datasets in total. We generated the features and stored five more feature sheets. Also, we did the statistical analysis and reduced the features using the RFE wrapper. We used the first 15 from the reduced features in this phase for every combination. Thus, we needed to filter the feature sheets and consider only those 15 features.

However, in this case, we selected the best three classifiers from the previous phase, and for these classifiers, we checked all the possible combinations. Then, we created the feature frequency histograms for those three classifiers, only when combined with the bootstrap as an evaluation method, to notice the most frequent features. Afterward, we chose the three networks that performed the best in the previous phase (with only the two categories) and utilized them in our DL system for the five categories. All the results can be noticed in the following subsection 4.1.3.

Chapter 4

Results

4.1 Detailed results presentation

Firstly, in Table 3 are presented, the dataset, the classes, and the number of the total patches.

Label	Category	Number of patches	Split in two categories
Healthy	Healthy	712	Benign -1303
Adenomatous	Benign	591	
Moderately differentiated	Grade I	869	Malignant -1891
Moderately to poorly differentiated	Grade II	414	
Poorly differentiated	Grade III	608	
		Total: 3194	

Table 3. Explanation of the dataset and number of patches for every category

4.1.1 Investigation

In Table 4, are displayed the results for our little investigation. The results highlighted in green are the best ones across all the results. The results highlighted in blue represent the second-best and third-best accuracies.

	LOO		Hold-out		K-Fold		Bootstrap	
Classifier	RFE	PCA	RFE	PCA	RFE	PCA	RFE	PCA
MDC	81.94	73.41	81.88	75.88	81.61	73.77	81.19	74.52
KNN	87.79	82.94	86.29	82.67	87.46	83.11	87.91	85.25
Bayes	80.94	82.27	80.67	83.08	81.61	82.27	79.73	82.80
LDA	83.61	83.78	83.79	83.83	83.44	84.28	83.97	83.37
LogReg	81.94	76.09	79.88	75.88	82.44	75.42	76.76	74.81
Perceptron	85.79	83.78	81.71	82.46	82.94	81.44	80.04	80.48
SVM	82.44	83.44	81.67	81.42	82.27	82.78	81.38	81.49
RF	87.46	84.62	85.00	82.71	86.97	84.11	90.63	88.72
CART	86.79	80.10	81.71	77.58	84.62	82.44	89.02	86.84
Mean accuracy	84.30	81.16	82.51	80.61	83.71	81.07	83.40	82.03

Table 4. Accuracies % - Investigation (per three combinations)

4.1.2 Two categories

After doing this investigation, we moved on to our actual data. We used the RFE wrapper for further feature reduction and bootstrap and K-fold for evaluation methods. Table 5 shows the outcomes of performing our ML procedure on our actual data. Here, the results highlighted in green show which classifiers performed the best.

	K-Fold			Bootstrap		
Classifier	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
MDC	79.39	78.20	80.20	79.43	77.53	80.75
KNN	84.71	80.51	87.61	85.88	82.78	87.97
Bayes	79.69	75.29	82.74	79.68	73.36	84.07
LDA	81.77	72.37	88.24	81.83	72.47	88.31
LogReg	68.64	29.78	95.45	65.59	19.04	98.15
Perceptron	78.98	82.04	76.87	76.64	78.48	75.37
SVM	78.92	63.55	89.52	76.94	53.91	92.47
RF	85.03	82.50	86.77	90.20	87.63	91.99
CART	79.98	75.98	82.74	87.93	85.35	89.71

Table 5. Accuracies (%) of Benign and Malignant categories considering twenty-two (22) features

Feature Frequency Histograms
Thereafter, we can see the feature frequency histograms in Figures 33 and 34.

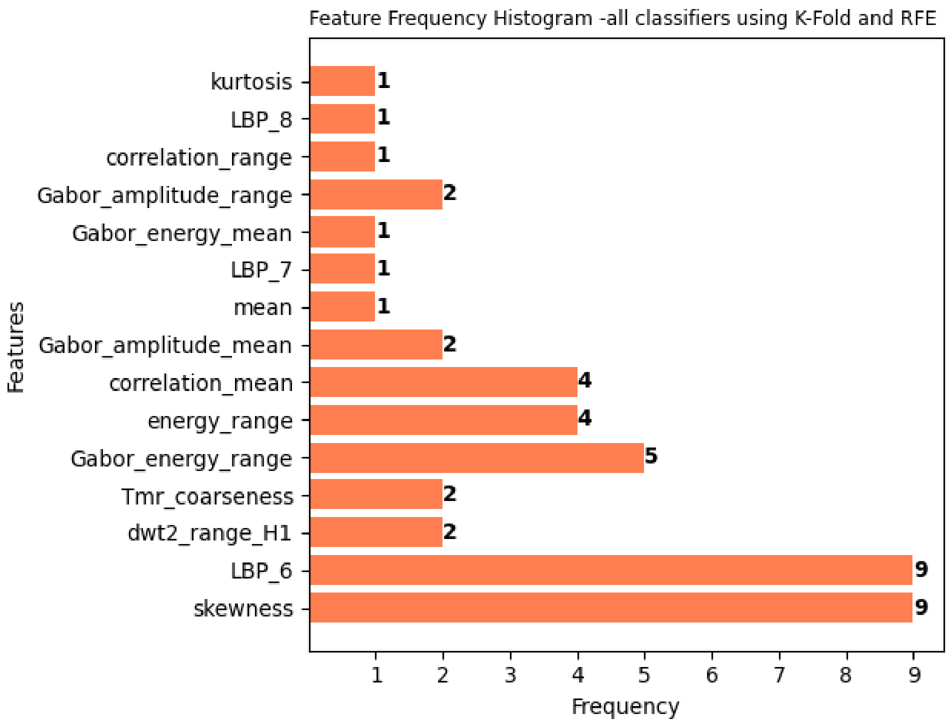


Figure 33. Feature Frequency Histogram, all classifiers K-Fold eval. method, RFE -feature

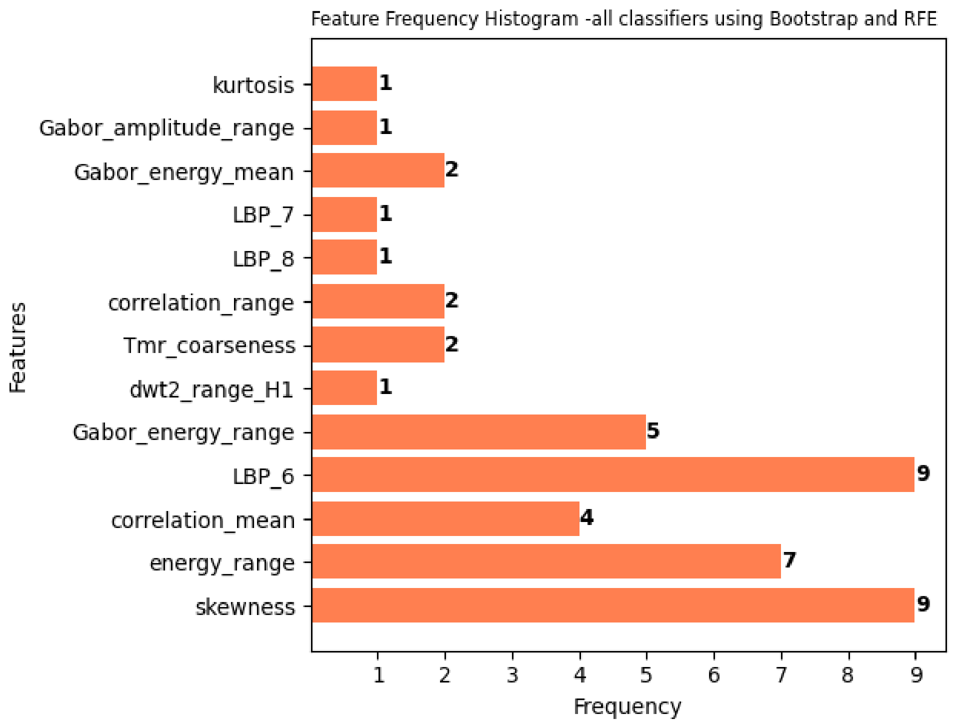


Figure 34. Feature Frequency Histogram, all classifiers, Bootstrap, RFE

Below, we can see the boxplot calculated during the statistical analysis of the five most frequent features encountered from the above histograms 33, 34.

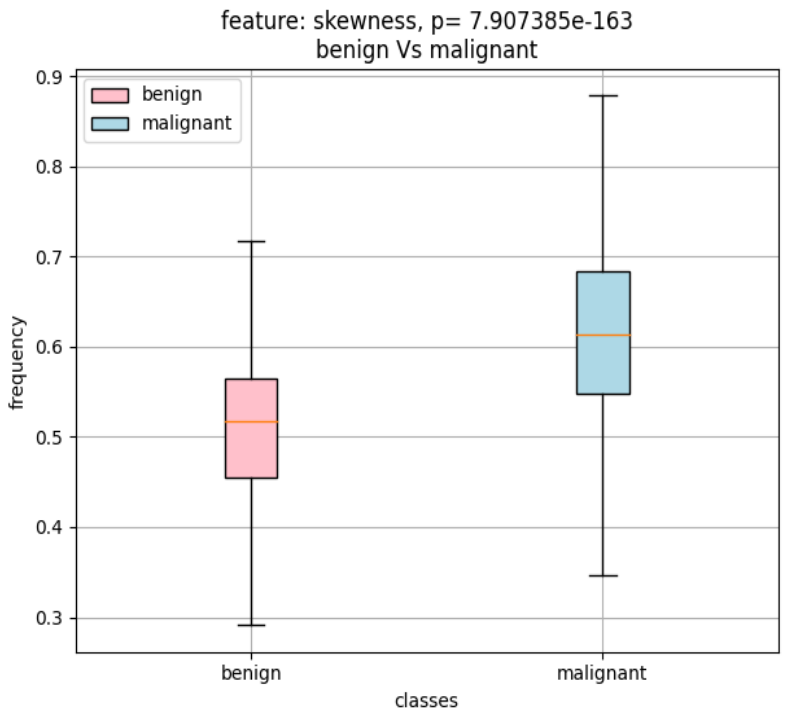


Figure 35. Boxplot showing the difference between benign and malignant for the feature: skewness

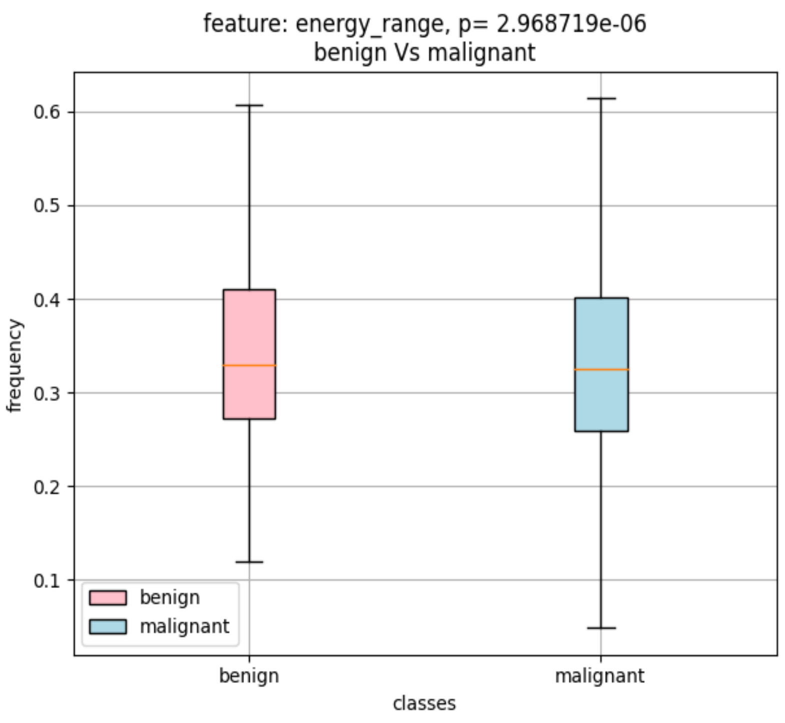


Figure 36. Boxplot showing the difference between benign and malignant for the feature: energy range

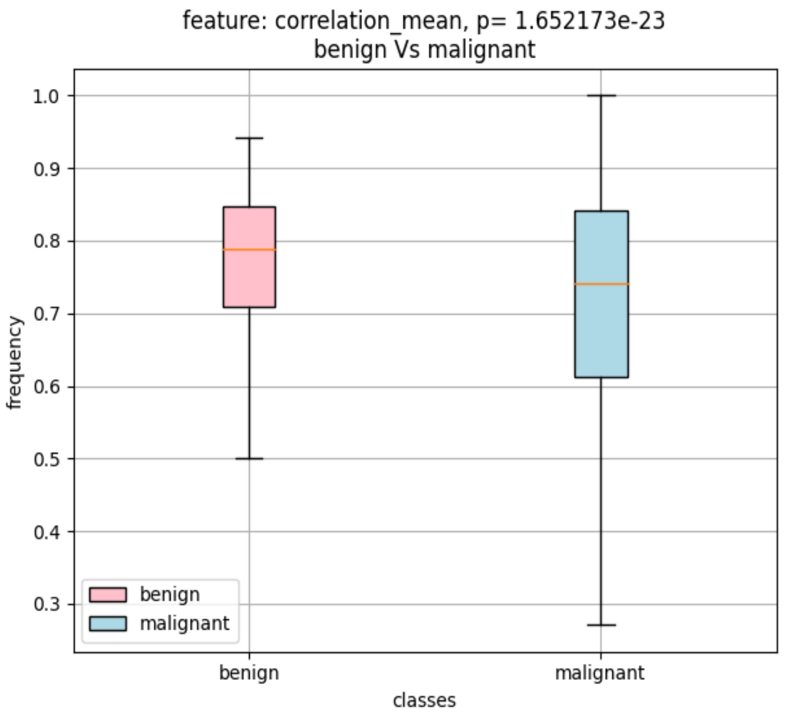


Figure 37. Boxplot showing the difference between benign and malignant for the feature: correlation mean

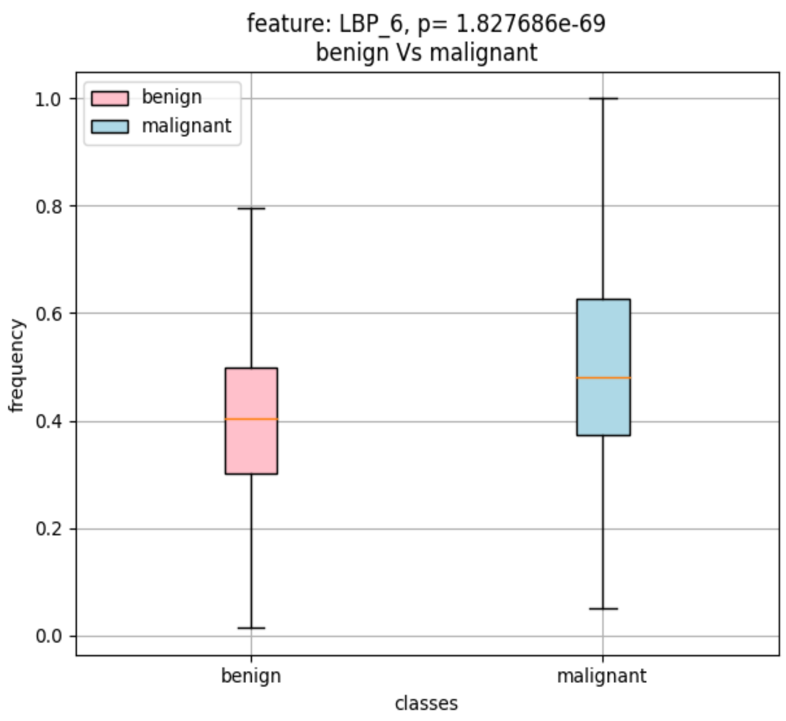


Figure 38. Boxplot showing the difference between benign and malignant for the feature: LBP6

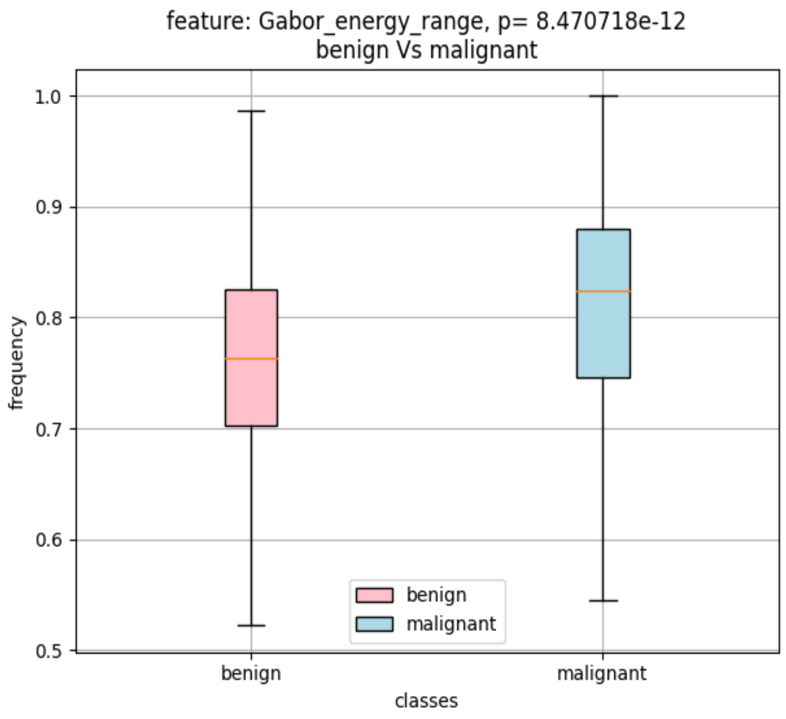


Figure 39. Boxplot showing the difference between benign and malignant for the feature: Gabor energy range

Table 6 presents the mean accuracies and the losses of 5 total runs of the networks we use for

20 and 100 epochs. Again, highlighted in green are the networks that perform the best and are chosen for the next phase with the five categories.

	20 Epochs		100 Epochs	
	Accuracies %	Loss	Accuracies %	Loss
CNN	84.66	0.452	86.23	1.164
VGG16	83.09	0.429	88.26	0.315
MNV2	87.79	0.311	88.42	0.294
RES50	89.52	0.267	90.92	0.254
INCV3	89.83	0.293	89.67	0.284
EFIB0	86.69	0.338	89.98	0.259

Table 6. Deep learning results for two categories: benign and malignant

4.1.3 Five categories

Moving on to the five categories, we followed the same methodology as the two categories, with some minor adjustments. First, having five different datasets (5 classes), we need to perform our procedure ten times to include all the combinations. Thus, we chose three of the nine classifiers in total. Table 7 displays the number of features that remain after applying the statistical tests. It is also important to mention that after the RFE wrapper reduces our features further, we select the first 15.

Class 1	**Class 2**	**Number of features**
Healthy	Benign/adenomatous	44
Healthy	Grade I	47
Healthy	Grade II	39
Healthy	Grade III	52
Benign	Grade I	44
Benign	Grade II	35
Benign	Grade III	46
Grade I	Grade II	13
Grade I	Grade III	39
Grade II	Grade III	46

Table 7. The numbers of features with statistical significance for each combination of the five classes

Tables 8, 9 show the accuracies of all ten combinations of the datasets using the KNN classifier with the RFE as feature reduction method and evaluation methods the K-Fold and the bootstrap. Again, Tables 12, 13 display the exact implementation, but for the Random Forest classifier and 10, 11 for the CART classifier.

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		80.63	88.92	90.04	87.71
Benign			89.71	79.76	84.63
Grade I				72.05	74.31
Grade II					75.49
Grade III					

Table 8. Accuracies (%) for five categories, using KNN classifier, K-Fold eval. method, and RFE

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		82.56	89.79	90.32	88.39
Benign			90.33	82.28	85.34
Grade I				76.67	78.09
Grade II					78.60
Grade III					

Table 9. Accuracies (%) for five categories, using KNN classifier, Bootstrap eval. method, and RFE

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		80.25	88.92	89.68	87.41
Benign			89.30	80.16	83.54
Grade I				74.08	74.71
Grade II					76.57
Grade III					

Table 10. Accuracies (%) for five categories, using RF classifier, K-Fold eval. method, and RFE

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		84.83	90.94	91.78	90.39
Benign			90.52	87.09	90.18
Grade I				79.39	84.06
Grade II					86.82
Grade III					

Table 11. Accuracies for five categories, using RF classifier, Bootstrap eval. method, and RFE

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		75.86	86.19	86.29	82.93
Benign			85.59	84.95	80.53
Grade I				69.09	76.75
Grade II					69.90
Grade III					

Table 12. Accuracies (%) for five categories, using CART classifier, KNN eval. method, and RFE

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		84.42	89.83	90.37	88.91
Benign			90.39	89.56	87.24
Grade I				80.47	85.13
Grade II					81.78
Grade III					

Table 13. Accuracies (%) for five categories, using CART classifier, Bootstrap eval. method, and RFE

Feature Frequency Histograms

We created the feature frequency histograms for the three classifiers but only when using the bootstrap evaluation method. These were constructed by considering the best feature combinations (the most frequent) across all ten class combinations. These can be viewed in the following images 41, 40, 42.

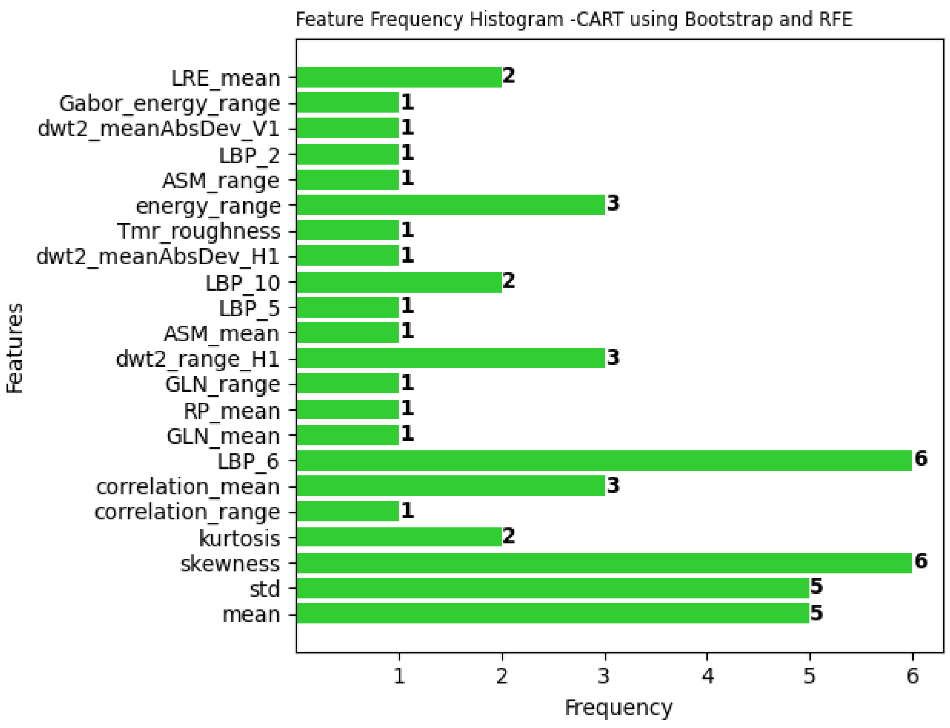


Figure 40. Feature Frequency Histogram, using CART classifier, Bootstrap, and RFE

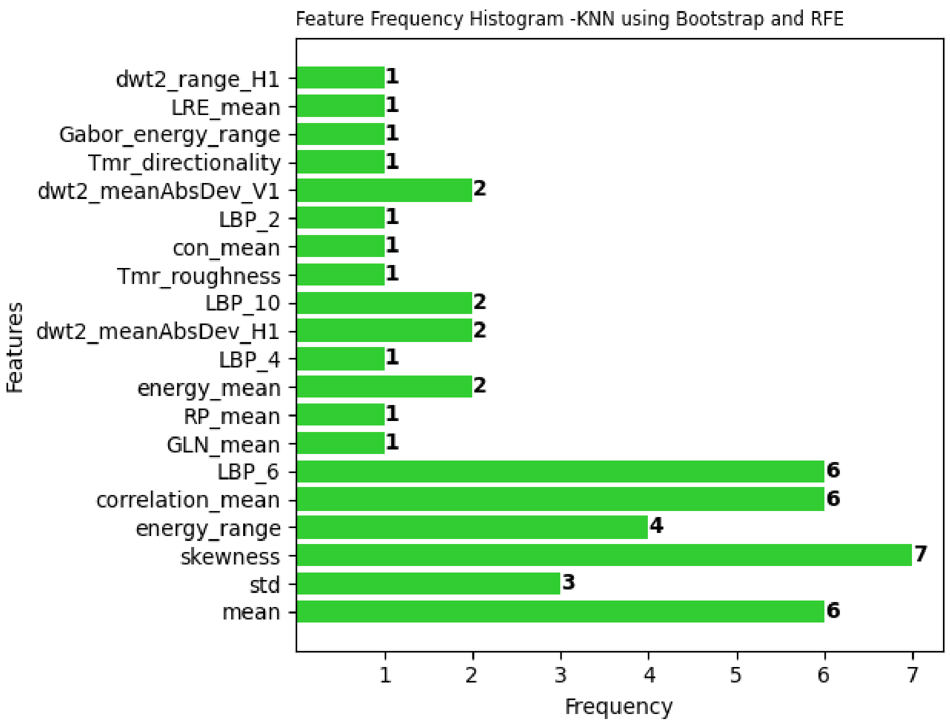


Figure 41. Feature Frequency Histogram, using KNN classifier, Bootstrap eval. method, and RFE

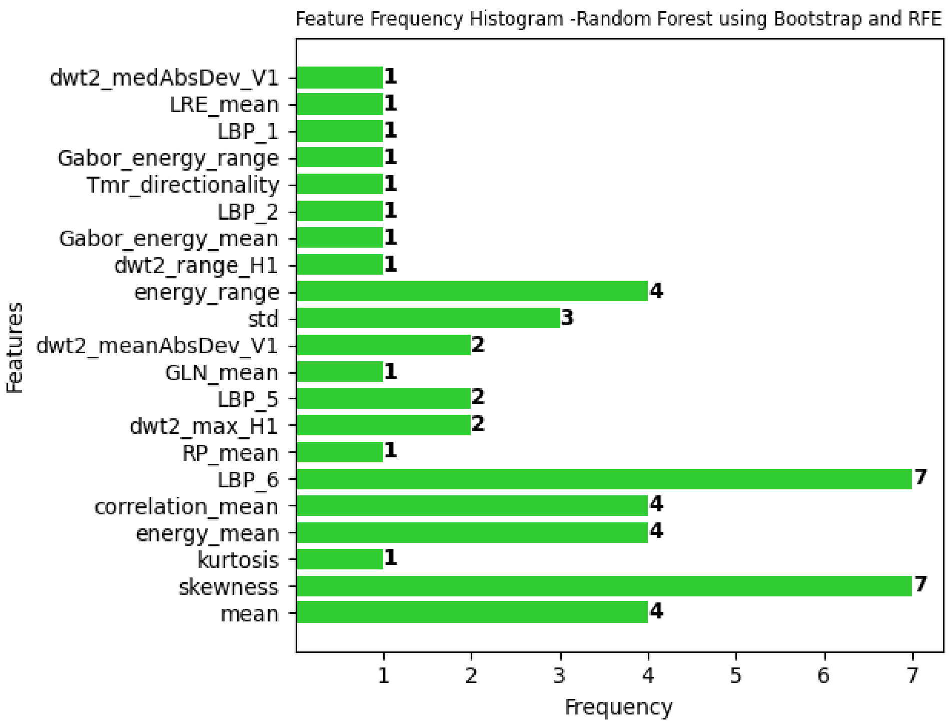


Figure 42. Feature Frequency Histogram using the RF classifier with Bootstrap and RFE

Finally, Tables 14, 15, 16 show the results for the three networks, which are the resnet, inception, and efficientnet, respectively, for 100 epochs.

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		83.14	93.69	90.71	89.77
Benign			90.07	92.54	93.33
Grade I				70.04	76.01
Grade II					73.65
Grade III					

Table 14. Accuracies (%) for five categories, using the RES50 with epochs=100

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		74.33	89.27	87.61	86.74
Benign			89.04	89.55	87.08
Grade I				68.48	69.93
Grade II					66.34
Grade III					

Table 15. Accuracies (%) for five categories, using the INCV3 with epochs=100

	Healthy	Benign	Grade I	Grade II	Grade III
Healthy		78.93	90.22	88.94	92.05
Benign			91.09	89.06	90.00
Grade I				71.98	80.74
Grade II					74.63
Grade III					

Table 16. Accuracies (%) for five categories, using the EFIB0 with epochs=100

4.2 Key findings/results

Some of the main/key findings and results of our research are presented below:

- Using **Bootstrap** as an evaluation method during the ML procedure, we noticed better results than using the **K-Fold** for most classifiers.
- From all the available classifiers (9), the ones that perform the best (with higher accuracies) are the KNN (85.88% - bootstrap), LDA (81.83% -bootstrap), CART (87.93% -bootstrap), and Random Forest (90.20% -bootstrap). These classifiers performed the best using both the bootstrap and the K-Fold evaluation methods. Also, their values of sensitivity and specificity are satisfactory.
- After creating the feature frequency histograms, taking into consideration the results of all the classifiers for combinations of five features, we observed that the following features were the most frequent: ”corr mean,” ”gabor energy range,” ”energy range,” ”LBP6”, and ”skewness.”
- By calculating the boxplot of the aforementioned features, we can also see that the two classes differ and draw some conclusions, especially for the correlation mean, the gabor energy range, the LBP6, and the skewness.
- During the DL procedure, we ran all the networks for 20 epochs and 100 epochs, and we noticed a slight increase in the accuracies with the 100 epochs.

- Also, from the total six networks, the three that achieved the best results (accuracies) were the RES50 (90.92% -100 epochs), the INCV3 (89.70% -100 epochs), and the EFIB0 (89.90% - 100 epochs).
- For every combination of two of the five categories, performing the statistical analysis and applying the RFE wrapper, we reduced the initial total number of 68 features a lot. It can be noticed in Table 7. Furthermore, it can be marked that the features that can be used to differentiate grade I from grade II are very few, 13 in total.
- We checked every combination using three classifiers: the KNN, the CART, and the RF, using two evaluation methods that we used for the two categories, the K-Fold, and the bootstrap. The results showed that the bootstrap performed better as well as it did for the merged classes (benign and malignant).
- The best results in classifying the patches into the correct category, using the first 13 features after the feature reduction with the RFE wrapper was applied, were noticed for the CART and the Random Forest classifiers with the bootstrap as an evaluation method.
- The accuracy of the healthy-grade III combination of classes is slightly lower than that of the healthy-grade II combination of classes (This can be observed in the results of all three classifiers). The same peculiarity can also be observed between the benign-grade I and benign-grade II combination of classes.
- The lowest accuracy is encountered between the grade I/II classes, with the following values: 76.67% -KNN, 79.39% -Random Forest, and 80.47% -CART.
- In the feature frequency histograms designed for those three classifiers using the bootstrap (evaluation method), the most frequent features are the skewness, the LBP6, the correlation mean, the mean, and the energy range (also the std and the dwt2 range H1 - these are for CART, and energy mean -for Random Forest).
- In the DL procedure for the five classes, the best results come from the **RES50** and the **EFIB0** networks for 100 epochs.
- Again, the fact that the accuracy of the healthy-grade III combination of classes is slightly lower than that of the healthy-grade II combination of classes is observed in the results.
- In this case, the lowest accuracies are seen between the grade I-II and grade II-III categories.

Chapter 5

Discussion of results

5.1 Discussion and Conclusion

Now, we are going to examine the findings mentioned above. The investigation was a quick test to show us which methods best fit our problem. So, our ultimate goal is first to find the best combination of classifier, feature reduction, and evaluation methods and then to discover the best (and the smallest in size) set of features that can differentiate any two classes (e.g benign/malignant, healthy/benign, healthy/grade III, etc.).

Our ML system's input is the features already generated from several images. In our case, we generated a considerable number of features, totaling 68. It is understandable to have that much variety, but the research can get more complex and complicated when we have too many features. It is getting harder to explain their importance and their correlation to each other. Thus, we must filter these features, discard the unimportant ones, and keep only those that bring value to our specific problem and research.

The first step to reduce the number of features was using statistical tests and keeping only the features with statistically significant differences. Then, by using the RFE wrapper with estimator the LogisticRegression and with the parameter *n_feature_to_select* set to default, which means selecting half the number of the initial number of features, we concluded with a total number of 22 most important features.

We utilized these twenty-two (22) features for our ML system; as a matter of fact, they were the system's input. The system first checks the combinations of two, three, four, and, lastly, five features and stores the combination of features that gives the best accuracy. Regarding the two evaluation methods that we used, the bootstrap performs slightly better than the K-Fold. As explained in theory, the K-fold method splits the dataset in K folds (10 in our case), and the model is trained ten times. The fact that it uses different subsets for training and validation makes it more reliable in evaluating the model's performance. On the other hand, the bootstrap method, which is also called sampling with replacement, selects random sample data and constructs many bootstrap datasets, which are used both in the training and evaluation phases of the model (for ten epochs/iterations in our case), and the results are averaged. The fact that it randomly selects samples indicates it might select some samples multiple times, which can introduce some bias in case there are outliers in the original dataset and those are included in the bootstrap datasets/samples. In general, both evaluation methods are preferred when the dataset is limited in size. Thus, the K-Fold might be more objective than the bootstrap; even though we notice a slight decrease in the accuracies for all the classifiers, we might consider choosing the K-Fold method for our future work. The classifier with the best performance is the Random Forest, reaching 85.03% with the K-Fold and 90.20% with the bootstrap evaluation method. Also, outstanding results demonstrated the KNN classifier with 85.88% and the CART classifier with 87.93% -using the bootstrap.

Moreover, the sensitivity and specificity values are metrics that estimate the performance and effectiveness of the model, especially in binary classification problems, like in our case. The values of these two metrics are satisfactory and comparable. For our best accuracy, the sensitivity is equal to 87.63% and the specificity 91.99% for our RF classifier (with bootstrap as the evaluation method). In our case, where we are investigating a medical problem, sensitivity is considerably more critical. We want to minimize the false negative cases (those that are malignant but were classified as benign) so that we do not miss a positive for malignancy (malignant) case and delay the diagnosis and treatment. Thus, we want our system to achieve high sensitivity, which, considering our results, happens. However, at the same time, we also want to achieve as high specificity as possible to avoid false negatives and false alarms, which will lead a potential patient to further unnecessary investigation. Also, the comparable values of those metrics, given that we have imbalanced datasets, indicate a balanced performance, meaning there is no bias towards one class (especially the numerous ones).

Continuing, we notice from the feature frequency histograms that five features were selected the most across all classifiers: skewness, energy range, correlation mean, LBP6, and Gabor energy range. We also noticed that these features existed in the top features selection for all the available feature combinations up to five. The first three features describe the patches statistically. The skewness belongs to the first-order statistics, while the energy range and correlation mean belong to the second-order statistics derived from the patches. The LPB6 belongs to the LPB (Local binary pattern) set of features, which are ten in total in our approach. The LBP features are trying to collect information about the texture of the images by using local patterns rather than examining every pixel of the image as it is done in the statistical approach (with the statistical features). Moreover, the Gabor energy range belongs to the Gabor set of features, which use the Gabor filters that mimic the human visual system. So we have features from almost all the different sets of features, describing and evaluating different aspects of the images, that makes our research more complete, holistic, and thorough.

Next, the comparison of the two boxplots can show us valuable insights into the distribution and central tendencies of the two datasets. We can see that the boxplots are the most distinguishable for the skewness, given that their boxes do not overlap and that the malignant class's median value is higher than the median of the benign class. This higher median value for malignant cases indicates that malignant cases tend to have higher skewness on average than benign ones. This difference in the median value placement is also observed for the LBP6, the correlation-mean, and the Gabor energy range. However, parts of the boxes overlap in those features, although the overlapping is not as substantial as in the boxplots for the "energy range" feature. It is essential to mention that the box of the boxplot itself demonstrates the spread of the middle 50% of the dataset. In addition to all of the above, the length of the boxes gives us an understanding of the variability of the feature; the more extended the box, the higher the variability. In the case of the "correlation mean" and "LBP6", we can see a much longer box for the malignant category, implying more increased variability. The increased variability in the malignant category suggests a greater diversity or heterogeneity in the malignant class, which could indicate different subgroups or types of malignant cases. Consequently, we can see from the boxplots that the features themselves can differentiate the two classes to a certain extent (we can spot differences and draw conclusions), and by utilizing them in combinations of two, three, four, or even all the five together, we can achieve excellent results.

For the DL approach, we used six networks, and the results for the performance of all of them were satisfactory. Firstly, we tested them using 20 epochs and noticed good results. Their losses were also less than 0.5. After that, we continued with the 100 epochs to discover if the accuracies were improving, also considering the values of the losses. For CNN, which is trained from scratch, although the accuracy increased, the loss number also increased, suggesting overfitting or other problems with the learning process. On the other hand, for all the rest of the networks, we noticed that they performed better, their accuracy increased slightly, and also, at the same time, their number of losses decreased. The fact that by increasing the epochs, we saw better results and lowered losses generally means the model/network benefits from this further training (the model continues to learn and improve its performance on the training data). Since the accuracies improved slightly, we (by increasing the epochs to 100) decided not to increase the number of epochs further. The top three are the resnet, the inception, and the efficientnet. The best for our problem is the resnet with an accuracy of 90.92% -for 100 epochs. These results are comparable with the results of our ML system. Thus, choosing to continue with the DL system is also a way to compare and evaluate our results from the ML system and point out differences/ similarities.

Noticing the promising results for our two merge classes, the benign and malignant, we thought we should proceed further with our research. Hence, we split the two classes into five. Moreover, we tested all five classes in combinations of two (10 combinations in total). We followed the same procedure as we did for the two classes. Nevertheless, in this case, after we did the statistical analysis and the further feature reduction with the RFE wrapper, we chose the first fifteen (15) features of the remaining. It was already computationally intensive to check for all these combinations of datasets, and we did want to become more intensive (and extensive) by choosing more than fifteen (15) features. After all, we reviewed only up to five combinations of features, and selecting more than fifteen (15) or a more increased number of feature combinations would not significantly affect our results. Sacrificing that much computational resources for a slight improvement at this point was not worth it.

An interesting finding is that we notice the least number of features (13) between grades I and II by observing the remaining features after applying the statistical tests. That means that very few features can be used to distinguish these two categories and that we expect to be the most challenging combination for our system to differentiate. For the ML procedure, for this part, we selected the best three classifiers from the previous phase: the KNN, CART, and the RF. We also used both evaluation methods, the K-Fold, and the bootstrap, and we noticed the exact same thing: using the bootstrap, we achieved better results than using the K-Fold. As we commented previously for two categories, the best results for the five categories were also accomplished using the RF for some combinations of datasets and the CART for the other combination of datasets with the bootstrap as an evaluation method. The two classifiers have no substantial differences in their results. By observing the tables, we can also verify that the most challenging combination to distinguish is grade I from grade II. This combination presented the lowest accuracy for all the classifiers using the two evaluation methods. Moreover, a peculiar thing is that the accuracy of the healthy-grade III combination of classes is slightly lower than that of the healthy-grade II combination of classes. This thing can be observed in the results of all three classifiers. We expected the accuracy of the healthy-grade III combination of classes to be higher than that of the healthy-grade II class combination. This peculiarity can be explained since our dataset is imbalanced, and the patches in the grade II category are fewer (414) than those in healthy (869) and grade III (608). So, it seems that the higher accuracy for the healthy-grade II combination of classes might be due to overfitting. The model, having limited samples, might memorize and learn certain cases and not learn the underlying patterns in the data. Thus, the accuracy of the healthy-grade II class combination is not that reliable. In contrast, the accuracy of the healthy-grade III class combination is more realistic and reliable, even though it is slightly lower.

From the feature frequency histograms created for every classifier, we observe that the skewness, LBP6, correlation mean, and energy range are again in the top (six) feature selections. The Gabor energy range is not one of the top features. It is essential to note that the mean and std features are also in the top selections in this phase.

Lastly, we performed the DL procedure for all five class combinations using the previous phase's top three networks: the RES50, INCV3, and EFIB0 -only for 100 epochs. For some combinations, the resnet performed better, and for others, the efficientnet. These two models performed the best. We can also confirm the findings during the ML process. The accuracies are similar, and our conclusions about the lowest accuracy and the difference in accuracy between these two class combinations, healthy-grade III and healthy-grade II, were proved again.

5.1.1 Conclusions

In conclusion, we developed the decision support system that can differentiate the benign and malignant categories with success. Using the RF classifier with the bootstrap as the evaluation method, we achieve an accuracy of 90.2%, with a sensitivity of 87.63% and 91.99% specificity. Also, for our particular situation from the feature frequency histograms, we conclude with five features selected the most across all classifiers: skewness, energy range, correlation mean, LBP6, and Gabor energy range. These results were compared with those from the DL pre-trained models, and it was confirmed that the DL models performed successfully as well, achieving 90.92% using the INCV3 model—with 100 epochs.

Things are more complicated for the five classes, mainly due to limited data and the imbalanced dataset. However, using the RF classifier with the bootstrap as the evaluation method, we achieve accuracies between 79.39% and 91.78%, across all the class combinations. The following features also appeared the most in the feature frequency histograms: skewness, energy range, correlation mean, and LBP6. Finally, regarding the DL procedure, the RES50 model performed the most satisfactorily, achieving accuracies between 70.04% and 93.69%, again across all the class combinations.

5.2 Future work

Some ideas and thoughts for future work are the following:

- First, we could find another more extensive dataset to test our current implementation. Also, we could experiment with data augmentation methods to grow the size of our current dataset (for all the different classes). This increase in size could improve the generalization of our models.

- Moreover, we must consider and handle the problem with the imbalanced datasets. We can add samples to the smaller-size dataset (for example, with augmentation) or take fewer samples from the more extensive dataset to equal the minority dataset. Alternatively, even find other ways to cope with that.

- Then, we could also examine ensemble methods to combine predictions of the models that performed the best for both ML and DL procedures —doing that, we might encounter better accuracies and overall performance.

Chapter 6

Appendix - List of Features

ID	Name	Abbreviation
1	Mean value	mean
2	Standard Deviation	std
3	Skewness	skewness
4	Kurtosis	kurtosis
5	Angular Second Moment Mean	asm_mean
6	Angular Second Moment Range	asm_range
7	Energy Mean	energy_mean
8	Energy Range	energy_range
9	Contrast Mean	contrast_mean
10	Contrast Range	contrast_range
11	Correlation Mean	correlation_mean
12	Correlation Range	correlation_range
13	Dissimilarity Mean	dissimilarity_mean
14	Dissimilarity Range	dissimilarity_range
15	Homogeneity Mean	homogeneity_mean
16	Homogeneity Range	homogeneity_range
17	Short Run Emphasis Mean	sre_mean
18	Short Run Emphasis Range	sre_range
19	Long Run Emphasis Mean	lre_mean
20	Long Run Emphasis Range	lre_range
21	Gray-Level Non-Uniformity Mean	glnu_mean
22	Gray-Level Non-Uniformity Range	glnu_range
23	Run Length Non-Uniformity Mean	rlnu_mean
24	Run Length Non-Uniformity Range	rlnu_range
25	Run Percentage Mean	rpc_mean
26	Run Percentage Range	rpc_range
27	Local binary pattern 1	LPB_1
28	Local binary pattern 2	LPB_2
29	Local binary pattern 3	LPB_3
30	Local binary pattern 4	LPB_4
31	Local binary pattern 5	LPB_5
32	Local binary pattern 6	LPB_6
33	Local binary pattern 7	LPB_7
34	Local binary pattern 8	LPB_8
35	Local binary pattern 9	LPB_9
36	Local binary pattern 10	LPB_10

ID	Name	Abbreviation
37	Discrete Wavelet Transformation mean Horizontal 1	dwt2_mean_H1
38	Discrete Wavelet Transformation median Horizontal 1	dwt2_median_H1
39	Discrete Wavelet Transformation mode Horizontal	dwt2_mode_H1
40	Discrete Wavelet Transformation max Horizontal	dwt2_max_H1
41	Discrete Wavelet Transformation min Horizontal	dwt2_min_H1
42	Discrete Wavelet Transformation range Horizontal	dwt2_range_H1
43	DWT median absolute deviation Horizontal 1	dwt2_medAbsDev_H1
44	DWT mean absolute deviation Horizontal 1	dwt2_meanAbsDev_H1
45	Discrete Wavelet Transformation mean Vertical 1	dwt2_mean_V1
46	Discrete Wavelet Transformation median Vertical 1	dwt2_median_V1
47	Discrete Wavelet Transformation mode Vertical 1	dwt2_mode_V1
48	Discrete Wavelet Transformation max Vertical 1	dwt2_max_V1
49	Discrete Wavelet Transformation min Vertical 1	dwt2_min_V1
50	Discrete Wavelet Transformation range Vertical 1	dwt2_range_V1
51	DWT median absolute deviation Vertical 1	dwt2_medAbsDev_V1
52	DWT mean absolute deviation Vertical 1	dwt2_meanAbsDev_V1
53	Discrete Wavelet Transformation mean Diagonal 1	dwt2_mean_D1
54	Discrete Wavelet Transformation median Diagonal 1	dwt2_median_D1
55	Discrete Wavelet Transformation mode Diagonal 1	dwt2_mode_D1
56	Discrete Wavelet Transformation max Diagonal 1	dwt2_max_D1
57	Discrete Wavelet Transformation min Diagonal 1	dwt2_min_D1
58	Discrete Wavelet Transformation range Diagonal 1	dwt2_range_D1
59	DWT median absolute deviation Diagonal 1	dwt2_medAbsDev_D1
60	DWT mean absolute deviation Diagonal 1	dwt2_meanAbsDev_D1
61	Tamura Coarseness	tmr_coarseness
62	Tamura Contrast	tmr_contrast
63	Tamura Directionality	tmr_directionality
64	Tamura Roughness	tmr_roughness
65	Gabor energy mean	Gabor_energy_mean
66	Gabor energy range	Gabor_energy_range
67	Gabor amplitude mean	Gabor_amplitude_mean
68	Gabor amplitude range	Gabor_amplitude_range

References

[1] *Cancer Today — gco.iarc.fr*. https://gco.iarc.fr/today/fact-sheets-cancers. [Accessed 04-03-2024].

[2] CA CO. “Facts & Figures 2011-2013.” In: (2011).

[3] Karen Simon. “Colorectal cancer development and advances in screening.” In: *Clinical interventions in aging* (2016), pp. 967–976.

[4] BABitA PAngtey, JAgAt Mohini KAul, and SABitA MiShrA. “histogenesis of muscularis mucosa and muscularis externa of stomach: A human foetal study.” In: *Journal of Clinical and Diagnostic Research: JCDR* 11.8 (2017), AC01.

[5] Roberto Labianca et al. “Colon cancer.” In: *Critical Reviews in Oncology/Hematology* 74.2 (2010), pp. 106–133. ISSN: 1040-8428. DOI: https://doi.org/10.1016/j.critrevonc.2010.01.010. URL: https://www.sciencedirect.com/science/article/pii/S1040842810000119.

[6] *Colorectal Cancer—Patient Version — cancer.gov*. https://www.cancer.gov/types/colorectal. [Accessed 05-03-2024].

[7] *Colon Cancer Treatment (PDQ®)–Patient Version — cancer.gov*. https://www.cancer.gov/types/colorectal/patient/colon-treatment-pdq. [Accessed 04-Jun-2023].

[8] IARC Working Group on the Evaluation of Cancer-Preventive Strategies. *Colorectal Cancer Screening*. Vol. 17. IARC Handbooks of Cancer Prevention. International Agency for Research on Cancer, 2019. ISBN: 978-92-832-3023-6. URL: https://publications.iarc.fr/Book-And-Report-Series/Iarc-Handbooks-Of-Cancer-Prevention/Colorectal-Cancer-Screening-2019.

[9] https://www.facebook.com/sag.micro/. *Parts of a microscope with functions and labeled diagram — microbenotes.com*. https://microbenotes.com/parts-of-a-microscope/. [Accessed 17-Jun-2023].

[10] ZhiFei Lai and Huifang Deng. “Medical Image Classification Based on Deep Features Extracted by Deep Model and Statistic Feature Fusion with Multilayer Perceptron.” In: *Computational Intelligence and Neuroscience* 2018 (2018).

[11] C. A. Glasbey and G. W. Horgan. *Image Analysis for the Biological Sciences*. USA: John Wiley & Sons, Inc., 1995. ISBN: 0471937266.

[12] Robert Haralick, Kalaivani Shanmugam, and Itshak Dinstein. “Haralick RM, Shanmuga K, Dinstein ITextural features for image classification. IEEE Trans Syst Man Cybern 3: 610-621.” In: *Systems, Man and Cybernetics, IEEE Transactions on* SMC3 (Dec. 1973), pp. 610–621. DOI: 10.1109/TSMC.1973.4309314.

[13] Guorong Wang, Zhiwei Wang, and Zhengyu Jin. “Application and Progress of Texture Analysis in the Therapeutic Effect Prediction and Prognosis of Neoadjuvant Chemoradiotherapy for Colorectal Cancer.” In: *Chinese Medical Sciences Journal* 34.1 (2019), pp. 45–50. ISSN: 1001-9294. DOI: https://doi.org/10.24920/003572. URL: https://www.sciencedirect.com/science/article/pii/S1001929419300203.

[14] Charles L. Perrin. “Numerical Recipes in Fortran 90: The Art of Scientific Computing, Second Edition, Volume 2 (3 CD-ROMs and Manual) By William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. Cambridge University Press: New York, 1996.” In: *Journal of the American Chemical Society* 119.37 (1997), pp. 8748–8748. DOI: 10.1021/ja965936f. eprint: https://doi.org/10.1021/ja965936f. URL: https://doi.org/10.1021/ja965936f.

[15] Mryka Hall-Beyer. *GLCM Texture: A Tutorial v. 3.0 March 2017*. 2017. DOI: 10.11575/PRISM/10182. URL: https://prism.ucalgary.ca/handle/1880/51900.

[16] L.-K. Soh and C. Tsatsoulis. “Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices.” In: *IEEE Transactions on Geoscience and Remote Sensing* 37.2 (1999), pp. 780–795. DOI: 10.1109/36.752194.

[17] A. Chu, C. M. Sehgal, and J. F. Greenleaf. “Use of gray value distribution of run lengths for texture analysis.” English (US). In: *Pattern Recognition Letters* 11.6 (June 1990), pp. 415–419. ISSN: 0167-8655. DOI: 10.1016/0167-8655(90)90112-F.

[18] Belur V. Dasarathy and Edwin B. Holder. “Image characterizations based on joint gray level—run length distributions.” In: *Pattern Recognition Letters* 12.8 (1991), pp. 497–502. ISSN: 0167-8655. DOI: https://doi.org/10.1016/0167-8655(91)80014-2. URL: https://www.sciencedirect.com/science/article/pii/0167865591800142.

[19] Topi Mäenpää and Matti Pietikäinen. “TEXTURE ANALYSIS WITH LOCAL BINARY PATTERNS.” In: *Handbook of Pattern Recognition and Computer Vision*, pp. 197–216. DOI: 10.1142/9789812775320_0011. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789812775320_0011. URL: https://www.worldscientific.com/doi/abs/10.1142/9789812775320_0011.

[20] Topi Mäenpää. “The local binary pattern approach to texture analysis - extensions and applications.” In: 2003.

[21] Olivier D. Faugeras and William K. Pratt. “Decorrelation Methods of Texture Feature Extraction.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-2.4 (1980), pp. 323–332. DOI: 10.1109/TPAMI.1980.4767031.

[22] J. Woods. “Two-dimensional discrete Markovian fields.” In: *IEEE Transactions on Information Theory* 18.2 (1972), pp. 232–240. DOI: 10.1109/TIT.1972.1054786.

[23] George R. Cross and Anil K. Jain. “Markov Random Field Texture Models.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-5.1 (1983), pp. 25–39. DOI: 10.1109/TPAMI.1983.4767341.

[24] A.K. Jain and K. Karu. “Learning texture discrimination masks.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.2 (1996), pp. 195–205. DOI: 10.1109/34.481543.

[25] Jacob Beck, K. Prazdny, and Azriel Rosenfeld. “A Theory of Textural Segmentation.” In: *Human and Machine Vision*. Ed. by Jacob Beck, Barbara Hope, and Azriel Rosenfeld. Notes and Reports in Computer Science and Applied Mathematics. Academic Press, 1983, pp. 1–38. DOI: https://doi.org/10.1016/B978-0-12-084320-6.50007-4. URL: https://www.sciencedirect.com/science/article/pii/B9780120843206500074.

[26] T. Ojala, M. Pietikainen, and T. Maenpaa. “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.7 (2002), pp. 971–987. DOI: 10.1109/TPAMI.2002.1017623.

[27] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. “Textural Features Corresponding to Visual Perception.” In: *IEEE Transactions on Systems, Man, and Cybernetics* 8.6 (1978), pp. 460–473. DOI: 10.1109/TSMC.1978.4309999.

[28] *VU Digital Repository: Application of Wavelet Transform and its Advantages Compared to Fourier Transform — inet.vidyasagar.ac.in*. http://inet.vidyasagar.ac.in:8080/jspui/handle/123456789/779. [Accessed 06-Jan-2023].

[29] Y. Meyer and S. Roques. *Progress in Wavelet Analysis and Applications: Proceedings of the International Conference ”Wavelets and Applications,” Toulouse, France - June 1992*. Editions Frontieres, 1993. ISBN: 9782863321300. URL: https://books.google.gr/books?id=aHux78oQbbkC.

[30] T. Chang and C.-C.J. Kuo. “Texture analysis and classification with tree-structured wavelet transform.” In: *IEEE Transactions on Image Processing* 2.4 (1993), pp. 429–441. DOI: 10.1109/83.242353.

[31] S Arivazhagan and L Ganesan. “Texture classification using wavelet transform.” In: *Pattern Recognition Letters* 24.9 (2003), pp. 1513–1521. ISSN: 0167-8655. DOI: https://doi.org/10.1016/S0167-8655(02)00390-2. URL: https://www.sciencedirect.com/science/article/pii/S0167865502003902.

[32] M. R. Turner. “Texture discrimination by Gabor functions.” In: *Biological Cybernetics* 55.2 (Nov. 1986), pp. 71–82. ISSN: 1432-0770. DOI: 10.1007/BF00341922. URL: https://doi.org/10.1007/BF00341922.

[33] Peter Howarth and Stefan Rüger. “Evaluation of Texture Features for Content-Based Image Retrieval.” In: *Image and Video Retrieval*. Ed. by Peter Enser, Yiannis Kompatsiaris, Noel E. O’Connor, Alan F. Smeaton, and Arnold W. M. Smeulders. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 326–334. ISBN: 978-3-540-27814-6.

[34] B.S. Manjunath and W.Y. Ma. “Texture features for browsing and retrieval of image data.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.8 (1996), pp. 837–842. DOI: 10.1109/34.531803.

[35] I. Fogel and D. Sagi. “Gabor filters as texture discriminator.” In: *Biological Cybernetics* 61.2 (June 1989), pp. 103–113. ISSN: 1432-0770. DOI: 10.1007/BF00204594. URL: https://doi.org/10.1007/BF00204594.

[36] Joel Grus. *Data Science from Scratch: First Principles with Python*. Beijing: O’Reilly, 2. ISBN: 978-1-4919-0142-7. URL: http://my.safaribooksonline.com/9781491901427.

[37] A. Burkov. *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019. ISBN: 9781999579517. URL: https://books.google.gr/books?id=0jbxwQEACAAJ.

[38] Daisuke Komura and Shumpei Ishikawa. “Machine Learning Methods for Histopathological Image Analysis.” In: *Computational and Structural Biotechnology Journal* 16 (2018), pp. 34–42. ISSN: 2001-0370. DOI: https://doi.org/10.1016/j.csbj.2018.01.001. URL: https://www.sciencedirect.com/science/article/pii/S2001037017300867.

[39] Korsuk Sirinukunwattana, David R. J. Snead, and Nasir M. Rajpoot. “A Stochastic Polygons Model for Glandular Structures in Colon Histology Images.” In: *IEEE Transactions on Medical Imaging* 34.11 (2015), pp. 2366–2378. DOI: 10.1109/TMI.2015.2433900.

[40] Rarasmaya Indraswari, Rika Rokhana, and Wiwiet Herulambang. “Melanoma image classification based on MobileNetV2 network.” In: *Procedia Computer Science* 197 (2022). Sixth Information Systems International Conference (ISICO 2021), pp. 198–207. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2021.12.132. URL: https://www.sciencedirect.com/science/article/pii/S1877050921023565.

[41] Johnson Kolluri, Vinay Kumar Kotte, MSB Phridviraj, and Shaik Razia. “Reducing overfitting problem in machine learning using novel L1/4 regularization method.” In: *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*. IEEE. 2020, pp. 934–938.

[42] Alok Sharma, Kuldip K. Paliwal, and Godfrey C. Onwubolu. “Class-dependent PCA, MDC and LDA: A combined classifier for pattern classification.” In: *Pattern Recognition* 39.7 (2006), pp. 1215–1229. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2006.02.001. URL: https://www.sciencedirect.com/science/article/pii/S0031320306000410.

[43] *A KNN Undersampling Approach for Data Balancing — scirp.org*. https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/journal/paperinformation.aspx?paperid=60996. [Accessed 07-Jan-2023].

[44] B.V. Dasarathy. *Nearest Neighbor (NN) Norms: Nn Pattern Classification Techniques*. IEEE Computer Society Press tutorial. IEEE Computer Society Press, 1991. ISBN: 9780818689307. URL: https://books.google.gr/books?id=pYNAmAEACAAJ.

[45] Harry Zhang. “The optimality of naive Bayes.” In: *Aa* 1.2 (2004), p. 3.

[46] Kenji Watanabe, Takumi Kobayashi, and Toshikazu Wada. “Semi-Supervised Feature Transformation for Tissue Image Classification.” In: *PLOS ONE* 11.12 (Dec. 2016), pp. 1–20. DOI: 10.1371/journal.pone.0166413. URL: https://doi.org/10.1371/journal.pone.0166413.

[47] Suresh Balakrishnama and Aravind Ganapathiraju. “Linear discriminant analysis-a brief tutorial.” In: *Institute for Signal and information Processing* 18.1998 (1998), pp. 1–8.

[48] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[49] Stephan Dreiseitl and Lucila Ohno-Machado. “Logistic regression and artificial neural network classification models: a methodology review.” In: *Journal of Biomedical Informatics* 35.5 (2002), pp. 352–359. ISSN: 1532-0464. DOI: https://doi.org/10.1016/S1532-0464(03)00034-0. URL: https://www.sciencedirect.com/science/article/pii/S1532046403000340.

[50] David G. Kleinbaum and Mitchel Klein. “Introduction to Logistic Regression.” In: *Logistic Regression: A Self-Learning Text*. New York, NY: Springer New York, 2010, pp. 1–39. ISBN: 978-1-4419-1742-3. DOI: 10.1007/978-1-4419-1742-3_1. URL: https://doi.org/10.1007/978-1-4419-1742-3_1.

[51] Michele D Estebon. “Perceptrons: An associative learning network.” In: *Virginia Tech.—1997* (1997).

[52] Alireza Khotanzad and J-H Lu. “Classification of invariant image representations using a neural network.” In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38.6 (1990), pp. 1028–1038.

[53] Javier M. Moguerza and Alberto Muñoz. “Support Vector Machines with Applications.” In: *Statistical Science* 21.3 (2006), pp. 322–336. ISSN: 08834237. URL: http://www.jstor.org/stable/27645765 (visited on 03/16/2023).

[54] Alka Rani, Nirmal Kumar, Jitendra Kumar, Jitendra Kumar, and Nishant K. Sinha. "Chapter 6 - Machine learning for soil moisture assessment." In: *Deep Learning for Sustainable Agriculture*. Ed. by Ramesh Chandra Poonia, Vijander Singh, and Soumya Ranjan Nayak. Cognitive Data Science in Sustainable Computing. Academic Press, 2022, pp. 143–168. ISBN: 978-0-323-85214-2. DOI: https://doi.org/10.1016/B978-0-323-85214-2.00001-X. URL: https://www.sciencedirect.com/science/article/pii/B978032385214200001X.

[55] Yan-Yan Song and LU Ying. "Decision tree methods: applications for classification and prediction." In: *Shanghai archives of psychiatry* 27.2 (2015), p. 130.

[56] Ioannis Mollas, Grigorios Tsoumakas, and Nick Bassiliades. "LionForests: Local Interpretation of Random Forests through Path Selection." In: (Nov. 2019).

[57] Leo Breiman. "Random forests." In: *Machine learning* 45.1 (2001), pp. 5–32.

[58] Isabelle Guyon and André Elisseeff. "An introduction to variable and feature selection." In: *Journal of machine learning research* 3.Mar (2003), pp. 1157–1182.

[59] Girish Chandrashekar and Ferat Sahin. "A survey on feature selection methods." In: *Computers & Electrical Engineering* 40.1 (2014). 40th-year commemorative issue, pp. 16–28. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2013.11.024. URL: https://www.sciencedirect.com/science/article/pii/S0045790613003066.

[60] Benson Mwangi, Tian Siva Tian, and Jair C. Soares. "A Review of Feature Reduction Techniques in Neuroimaging." In: *Neuroinformatics* 12.2 (Apr. 2014), pp. 229–244. ISSN: 1559-0089. DOI: 10.1007/s12021-013-9204-3. URL: https://doi.org/10.1007/s12021-013-9204-3.

[61] Baoqiang Wang, Yuan Wei, Shulin Liu, Dan Gu, and Dongfang Zhao. "Intelligent chatter detection for CNC machine based on RFE multi-feature selection strategy." In: *Measurement Science and Technology* 32.9 (June 2021), p. 095904. DOI: 10.1088/1361-6501/ac04e0. URL: https://dx.doi.org/10.1088/1361-6501/ac04e0.

[62] Sergios Theodoridis and Konstantinos Koutroumbas. "Chapter 10 - Supervised Learning: The Epilogue." In: *Pattern Recognition (Fourth Edition)*. Ed. by Sergios Theodoridis and Konstantinos Koutroumbas. Fourth Edition. Boston: Academic Press, 2009, pp. 567–594. ISBN: 978-1-59749-272-0. DOI: https://doi.org/10.1016/B978-1-59749-272-0.50012-8. URL: https://www.sciencedirect.com/science/article/pii/B9781597492720500128.

[63] AJ Larner. *The 2x2 matrix: contingency, confusion and the metrics of binary classification*. Springer Nature, 2024.

[64] C. Radhakrishna Rao. "R. A. Fisher: The Founder of Modern Statistics." In: *Statistical Science* 7.1 (1992), pp. 34–48. ISSN: 08834237. URL: http://www.jstor.org/stable/2245989 (visited on 03/05/2024).

[65] Basant Lal Agarwal. *Basic statistics*. New Age International, 2006.

[66] *Box Plot — six-sigma-material.com*. https://www.six-sigma-material.com/Box-Plot.html. [Accessed 03-01-2024].

[67] Shirley Dowdy, Stanley Wearden, and Daniel Chilko. *Statistics for research*. John Wiley & Sons, 2011.

[68] Matthew S Thiese, Brenden Ronna, and Ulrike Ott. "P value interpretations and considerations." en. In: *J Thorac Dis* 8.9 (Sept. 2016), E928–E931.

[69] Estibaliz Gómez-de-Mariscal, Vanesa Guerrero, Alexandra Sneider, Hasini Jayatilaka, Jude M. Phillip, Denis Wirtz, and Arrate Muñoz-Barrutia. "Use of the p-values as a size-dependent function to address practical differences when analyzing large datasets." In: *Scientific Reports* 11.1 (Oct. 2021), p. 20942. ISSN: 2045-2322. DOI: 10.1038/s41598-021-00199-5. URL: https://doi.org/10.1038/s41598-021-00199-5.

[70] Prabhaker Mishra, Uttam Singh, Chandra M Pandey, Priyadarshni Mishra, and Gaurav Pandey. "Application of student's t-test, analysis of variance, and covariance." en. In: *Ann Card Anaesth* 22.4 (Oct. 2019), pp. 407–411.

[71] Francois Chollet. *Deep learning with Python*. Simon and Schuster, 2021.

[72] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects." In: *IEEE Transactions on Neural Networks and Learning Systems* 33.12 (2022), pp. 6999–7019. DOI: 10.1109/TNNLS.2021.3084827.

[73] Çağrı Kaymak and Ayşegül Uçar. "A brief survey and an application of semantic image segmentation for autonomous driving." In: *Handbook of deep learning applications* (2019), pp. 161–200.

[74] Haoyan Yang, Jiangong Ni, Jiyue Gao, Zhongzhi Han, and Tao Luan. "A novel method for peanut variety identification and classification by Improved VGG16." In: *Scientific Reports* 11.1 (2021), pp. 1–17.

[75] Arohan Ajit, Koustav Acharya, and Abhishek Samanta. "A Review of Convolutional Neural Networks." In: *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. 2020, pp. 1–5. DOI: 10.1109/ic-ETITE47903.2020.049.

[76] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "Mobilenetv2: Inverted residuals and linear bottlenecks." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[77] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." In: *arXiv preprint arXiv:1704.04861* (2017).

[78] Bishwas Mandal, Adaeze Okeukwu, and Yihong Theis. "Masked face recognition using resnet-50." In: *arXiv preprint arXiv:2104.08997* (2021).

[79] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[80] Md. Belal Hossain, S.M. Hasan Sazzad Iqbal, Md. Monirul Islam, Md. Nasim Akhtar, and Iqbal H. Sarker. "Transfer learning with fine-tuned deep CNN ResNet50 model for classifying COVID-19 from chest X-ray images." In: *Informatics in Medicine Unlocked* 30 (2022), p. 100916. ISSN: 2352-9148. DOI: https://doi.org/10.1016/j.imu.2022.100916. URL: https://www.sciencedirect.com/science/article/pii/S235291482200065X.

[81] N. Dong, L. Zhao, C.H. Wu, and J.F. Chang. "Inception v3 based cervical cell classification combined with artificially extracted features." In: *Applied Soft Computing* 93 (2020), p. 106311. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.106311. URL: https://www.sciencedirect.com/science/article/pii/S1568494620302519.

[82] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

[83] Chunmian Lin, Lin Li, Wenting Luo, Kelvin C. P. Wang, and Jiangang Guo. "Transfer Learning Based Traffic Sign Recognition Using Inception-v3 Model." In: *Periodica Polytechnica Transportation Engineering* 47.3 (2019), pp. 242–250. DOI: 10.3311/PPtr.11480. URL: https://pp.bme.hu/tr/article/view/11480.

[84] Tashin Ahmed and Noor Hossain Nuri Sabab. "Classification and understanding of cloud structures via satellite images with EfficientUNet." In: *SN Computer Science* 3 (2022), pp. 1–11.

[85] Korsuk Sirinukunwattana, Josien PW Pluim, Hao Chen, Xiaojuan Qi, Pheng-Ann Heng, Yun Bo Guo, Li Yang Wang, Bogdan J Matuszewski, Elia Bruni, Urko Sanchez, et al. "Gland segmentation in colon histology images: The glas challenge contest." In: *Medical image analysis* 35 (2017), pp. 489–502.

[86] Devvi Sarwinda, Radifa Hilya Paradisa, Alhadi Bustamam, and Pinkie Anggia. "Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer." In: *Procedia Computer Science* 179 (2021). 5th International Conference on Computer Science and Computational Intelligence 2020, pp. 423–431. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2021.01.025. URL: https://www.sciencedirect.com/science/article/pii/S1877050921000284.

[87] Francesco Ponzio, Enrico Macii, Elisa Ficarra, and Santa Di Cataldo. "Colorectal cancer classification using deep convolutional networks." In: *Proceedings of the 11th international joint conference on biomedical engineering systems and technologies*. Vol. 2. 2018, pp. 58–66.

[88] Naresh Kumar, Manoj Sharma, Vijay Pal Singh, Charanjeet Madan, and Seema Mehandia. "An empirical study of handcrafted and dense feature extraction techniques for lung and colon cancer classification from histopathological images." In: *Biomedical Signal Processing and Control* 75 (2022), p. 103596. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2022.103596. URL: https://www.sciencedirect.com/science/article/pii/S1746809422001185.

[89] K. Jarrod Millman and Michael Aivazis. "Python for Scientists and Engineers." In: *Computing in Science & Engineering* 13.2 (2011), pp. 9–12. DOI: 10.1109/MCSE.2011.36.

[90] Alexander Mordvintsev and K Abid. "Opencv-python tutorials documentation." In: *Obtenido de https://media. readthedocs. org/pdf/opencv-python-tutroals/latest/opencv-python-tutroals. pdf* (2014).

[91] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python." In: *Nature methods* 17.3 (2020), pp. 261–272.

[92] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. "Scikit-learn: Machine learning in Python." In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[93] Sandro Tosi. *Matplotlib for Python developers*. Packt Publishing Ltd, 2009.

[94] Jason Brownlee. *Deep learning with Python: develop deep learning models on Theano and TensorFlow using Keras*. Machine Learning Mastery, 2016.

[95] Korsuk Sirinukunwattana et al. “Gland segmentation in colon histology images: The glas challenge contest.” In: *Medical Image Analysis* 35 (2017), pp. 489–502. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2016.08.008. URL: https://www.sciencedirect.com/science/article/pii/S1361841516301542.

[96] *Traditional-Feature-Extraction/Gray Level Run Length Matrix.py at main · Rohit-Kundu/Traditional-Feature-Extraction — github.com*. https://github.com/Rohit-Kundu/Traditional-Feature-Extraction/blob/main/Gray%20Level%20Run%20Length%20Matrix.py. [Accessed 28-02-2024].

[97] *Local Binary Pattern for texture classification &x2014; skimage 0.22.0 documentation — scikit-image.org*. https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_local_binary_pattern.html#sphx-glr-auto-examples-features-detection-plot-local-binary-pattern-py. [Accessed 13-01-2024].

[98] *Traditional-Feature-Extraction/Tamura.py at main · Rohit-Kundu/Traditional-Feature-Extraction — github.com*. https://github.com/Rohit-Kundu/Traditional-Feature-Extraction/blob/main/Tamura.py. [Accessed 19-02-2024].

[99] Filip Wasilewski. *2D Forward and Inverse Discrete Wavelet Transform &x2014; PyWavelets Documentation — pywavelets.readthedocs.io*. https://pywavelets.readthedocs.io/en/latest/ref/2d-dwt-and-idwt.html. [Accessed 13-01-2024].

[100] *Traditional-Feature-Extraction/Gabor.py at main · Rohit-Kundu/Traditional-Feature-Extraction — github.com*. https://github.com/Rohit-Kundu/Traditional-Feature-Extraction/blob/main/Gabor.py. [Accessed 28-02-2024].

Additional Bibliography

[101] "Representation." In: *Fundamentals of Digital Image Processing*. John Wiley & Sons, Ltd, 2010. Chap. 1, pp. 1–19. ISBN: 9780470689776. DOI: https://doi.org/10.1002/9780470689776.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470689776.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470689776.ch1.

[102] "Introduction." In: *Medical Image Analysis*. John Wiley & Sons, Ltd, 2011. Chap. 1, pp. 1–22. ISBN: 9780470918548. DOI: https://doi.org/10.1002/9780470918548.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470918548.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470918548.ch1.

[103] Joshua J Levy and A James O'Malley. "Don't dismiss logistic regression: the case for sensible extraction of interactions in the era of machine learning." In: *BMC medical research methodology* 20.1 (2020), pp. 1–15.

[104] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "MobileNetV2: Inverted Residuals and Linear Bottlenecks." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

[105] Pedro Silva, Eduardo Luz, Guilherme Silva, Gladston Moreira, Rodrigo Silva, Diego Lucio, and David Menotti. "COVID-19 detection in CT images with deep learning: A voting-based scheme and cross-datasets analysis." In: *Informatics in Medicine Unlocked* 20 (2020), p. 100427. ISSN: 2352-9148. DOI: https://doi.org/10.1016/j.imu.2020.100427. URL: https://www.sciencedirect.com/science/article/pii/S2352914820305773.

[106] Andreas Maier, Stefan Steidl, Vincent Christlein, and Joachim Hornegger. "Medical imaging systems: An introductory guide." In: (2018).

[107] Polina Golland and Bruce Fischl. "Permutation Tests for Classification: Towards Statistical Significance in Image-Based Studies." In: *Information Processing in Medical Imaging*. Ed. by Chris Taylor and J. Alison Noble. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 330–341. ISBN: 978-3-540-45087-0.

[108] Jonathan J Shuster. "Student t-tests for potentially abnormal data." In: *Statistics in medicine* 28.16 (2009), pp. 2170–2184.

[109] *skimage.feature &x2014; skimage 0.22.0 documentation — scikit-image.org*. https://scikit-image.org/docs/stable/api/skimage.feature.html#skimage.feature.graycoprops. [Accessed 12-01-2024].

[110] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.