Bachelor thesis

Classification of DPLDs in HRCT scans: A Comparative Study of Texture Analysis Methods and a Novel Statistical and Graph-Based Approach

by **Álvaro Amorim de Albuquerque**

advised by
Prof. Ph.D. Fabiane da Silva Queiroz

Federal University of Alagoas
Computing Institute
Maceió, Alagoas
November 29, 2023

FEDERAL UNIVERSITY OF ALAGOAS
Computing Institute

CLASSIFICATION OF DPLDS IN HRCT SCANS: A COMPARATIVE STUDY OF TEXTURE ANALYSIS METHODS AND A NOVEL STATISTICAL AND GRAPH-BASED APPROACH

Bachelor Thesis submitted to the Computing Institute from Federal University of Alagoas as a partial requirement to obtain the bachelor degree in Computer Engineering.

Álvaro Amorim de Albuquerque

Advisor: Prof. Ph.D. Fabiane da Silva Queiroz

Examining Board:

André Luiz Lins de Aquino	Prof. Ph.D., UFAL
Rian Gabriel Santos Pinheiro	Prof. Ph.D., UFAL

Maceió, Alagoas
November 29, 2023

Catalogação na Fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecário: Marcelino de Carvalho Freitas Neto – CRB-4 - 1767

O48i Albuquerque, Alvaro Amorim de.

Classification of DPLDs in HRCT scans : a comparative study of texture analysis methods and a novel statistical and graph-based approach / Alvaro Amorim de Albuquerque. – 2023.

59 f. : il.

Orientadora: Fabiane da Silva Queiroz.

Monografia (Trabalho de conclusão de curso em Engenharia de Computação) - Universidade Federal de Alagoas, Instituto de Computação. Maceió, 2023.

Bibliografia: f. 55-59.

1. Classificação de imagens. 2. Classificação de textura. 3. Descritor de textura (Teoria dos grafos). 4. Imagens médicas. 5. Doenças Pulmonares intersticiais. I. Título.

CDU: 004:616.24

Acknowledgments

I would like to thank my family for all the support I've had during this journey. Without their encouragement and faith in my potential, I would not be here and would not be getting this degree.

I would like to thank Prof. Fabi Queiroz for being my advisor in this thesis and my mentor in the academic field. Thanks to Prof. André "Alla" Aquino, who recommended me as a mentee to Prof. Fabi and provided help and guidance on our research. Also, thanks to Prof. Alla and Prof. Rian Pinheiro for your invaluable contribution as a member of the examination board for my thesis.

I would like to express my heartfelt gratitude to the colleagues and friends I've made along my path who played a crucial role to the completion of this graduation. Thanks for all the hard work and enjoyable moments we had. Without you all, this would have been much harder.

To everybody who played a part in this, my most sincere thanks.

Álvaro Amorim de Albuquerque

I guess I gotta go. I guess it's time to go.
Solána Imani Rowe

Abstract

Problems of texture classification are consistently challenging once the patterns of different instances can be very similar. Moreover, the descriptors need to be invariant to rotations, scale, and lighting variations. In the context of medical imaging, this group of methods can aid in diagnosing patients as part of the concept of Computer-Aided Diagnosis (CAD). In this paper, we review methods for texture classification in the context of classifying Diffuse Parenchymal Lung Diseases (DPLDs) on High-Resolution Computed Tomographies (HRCTs) and propose a new method that uses concepts of complex networks and statistical metrics. Our approach is based on mapping the input image into multiscale graphs and extracting the closeness centrality metric. We transform the multiscale closeness centrality images into one matrix that encapsulates local and global texture information. From the matrix, we extract a feature vector that represents a DPLD pattern. This vector is then combined with Haralick and Local Binary Pattern descriptors to generate the final feature vector. Once this process characterizes all the images, we go through a classification step to recognize the image. We analyze the proposed approach's performance by comparing it with other texture analysis methods and discussing its metrics for each class (DPLD pattern) of the dataset. After the evaluation of different methods and our proposed method, it is possible to conclude the effectiveness of our approach to aid the diagnose process of DPDLs. Furthermore, we can highlight our technique as an aid on the problem of diagnosing patients with COVID-19.

Keywords*: *Image Classification*; *Texture Classification*; *Graph-based Texture Descriptor*; *Medical Image Analysis*; *Diffuse Parenchymal Lung Diseases Classification.

Resumo

Problemas de classificação de textura são consistentemente desafiadores uma vez que padrões de diferentes instâncias podem ser bastante similares. Além disso, os descritores precisam ser invariantes a rotações e variações de escala e iluminação. No contexto de imagens médicas, esse grupo de métodos pode auxiliar no diagnóstico de pacientes como parte do conceito de Diagnóstico Auxiliado por Computador (Computer-aided Diagnosis - CAD). Neste paper, nós revisamos métodos de classificação de textura no contexto de classificação de Doenças Pulmonares Parenquimatosas Difusas (Diffuse Parenchymal Lung Diseases - DPLDs) em Tomografias Computadorizadas de Alta Resolução (High-Resolution Computed Tomographies - HRCTs) e propomos um novo método que utiliza conceitos de Redes Complexas e métricas estatísticas. Nosso método é baseado em mapear a imagem de entrada em grafos multi-escala e extrair métrica de centralidade de proximidade. Transformamos as imagens de centralidade de proximidade multi-escala em uma matriz que encapsula informação de textura global e local. Desta matriz, extraímos um vetor de features que representa um padrão de DPLD que, por sua vez, é combinado com descritores de Haralick e Padrão Binário Local (LBP) para gerar o vetor de features final. Uma vez que isto caracteriza todas as imagens, passamos para a etapa de classificação para reconhecer a imagem. Analisamos a performance do método proposto ao compará-lo com outros métodos de classificação de textura e discutindo as métricas para cada classe (padrão da DPLD) do dataset. Após a avaliação de diferentes métodos e do método proposto, é possível concluir a efetividade da nossa abordagem para auxiliar o diagnóstico de DPLDs. Além disso, podemos destacar nossa técnica como um auxílio ao problema de diagnosticar pacientes com COVID-19.

Palavras-chave*: *Classificação de Imagens*; *Classificação de Textura*; *Descritor de Textura Baseado em Grafos*; *Análise de Imagens Médicas*; *Classificação de Doenças Pulmonares Parenquimatosas Difusas.

List of Figures

List of Tables

Notations and Symbols

$\in$	belong to
$\subseteq$	subset
$M \times N$	if $M, N \in \mathbb{R}$, size of matrix of M rows and N columns
	if $M, N \subseteq \mathbb{R}$, Cartesian product of the sets M and N
$\approx$	approximately equal
$I(x, y)$	pixel at spatial location x and y of image I
$P(x, y)$	(x, y) entry of matrix P
$\mathbb{N}^{a \times b}$	set of natural matrices of a rows and b columns
$\mathbb{Q}^{a \times b}$	set of rational matrices of a rows and b columns
$\mathbb{R}^{a \cdot b}$, $\mathbb{R}^{ab}$	set of real vectors of length $a \cdot b$
$\mathbf{V}^T$	transpose of matrix V
$ROR(P, i)$	circular bit-wise right shift on P by i bits
min	minimum
max	maximum
$*$	convolution
$E(\cdot)$	expectation operator
u_i	i-th node in graph
e_{u_i,u_j}, w_{u_i,u_j}	weight of the edge that connects u_i and u_j
k	degree or valency
k_u	degree of node u
${k_u}^{in}$	in-degree of node u
${k_u}^{out}$	out-degree of node u
$CC(u)$	closeness centrality metric of u
d_{u_i,u_j}	length of the shortest path between vertices u_i and u_j
$dist(p_i, p_j)$	spatial distance of pixels p_i and p_j
$\alpha(u_i)$	activity of node u_i
$\frac{\partial E}{\partial w}$	partial derivative of E with respect to w

List of Acronyms

CAD	Computer-Aided Diagnosis
DPLD	Diffuse Parenchymal Lung Disease
ROI	Region Of Interest
HRCT	High-Resolution Computed Tomography
CT	Computed Tomography
COVID	Corona Virus Disease
CV	Computer Vision
GLCM	Gray Level Co-occurrence Matrix
HOG	Histogram of Oriented Gradient
LBP	Local Binary Pattern
CN	Complex Network
DWT	Discrete Wavelet Transform
IDWT	Inverse Discrete Wavelet Transform
AC	Approximation Coefficient
DC	Detail Coefficient
SWT	Stationary Wavelet Transform
IDV	Intensity Difference Vector
MF	Multiresolution Fractal
PCA	Principal Component Analysis
DL	Deep Learning
ML	Machine Learning
CNN	Convolutional Neural Network
DXA, DEXA	Dual-Energy X-ray Absorptiometry
BMD	Bone Mineral Density
OCT	Optical Coherence Tomography
MRI	Magnetic Resonance Imaging
MS	Multiple Sclerosis
T-CNN	Texture Convolutional Neural Network
FD	Fractal Dimension
SP	Shortest Paths
DN	Diffusion in Networks

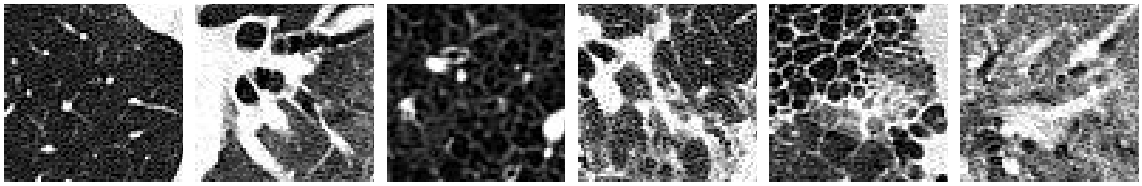
Contents

Chapter 1

Introduction

Diffuse Parenchymal Lung Diseases (DPLDs) form a group with more than 150 different pathologies that affect the interstitial region, including walls of the air sacs of the lungs and areas around blood vessels and lower airways [Pereyra et al., 2014]. The patient's complete history (symptoms, family history, disease record), physical examination, laboratory tests, pulmonary function tests, and visual findings on chest radiographs are essential in diagnosing DPLDs.

There are various image patterns associated with each type of DPLD. They denote different DPLDs with different histological appearances and clinical manifestations. Figure 1.1 illustrates visual texture patterns found in ROIs (Regions of Interest) of High-Resolution Computed Tomography (HRCT) images of the lung. These patterns vary from a healthy lung (Figure 1.1a) to five types of DPLDs: Pulmonary Consolidation (Figure 1.1b), Emphysematous Area (Figure 1.1c), Septal Thickening (Figure 1.1d), Honeycomb (Figure 1.1e), and Ground-glass Opacity (Figure 1.1f).



(a) Healthy (b) PC (c) EA (d) ST (e) HC (f) GGO

Figure 1.1: ROIs presenting texture patterns of DPLDs in high-resolution CT images: (a) Healthy Lung, (b) Pulmonary Consolidation (PC), (c) Emphysematous Area (EA), (d) Septal Thickening (ST), (e) Honeycomb (HC) and (f) Ground-glass Opacity (GGO) (Adapted from [Pereyra et al., 2014]).

DPLDs occasionally mimic each other, either because they share identical HRCT findings or because of the layover of patterns. In this context and with the increase use of the technology as an aid to recognise patterns and classify objects, Computer-Aided Diagnosis (CAD) has become one of the major research subjects in medical imaging and diagnostic radiology [Mori, 2020].

Medical imaging analysis is a field of study that has been gaining much attention recently. It consists of using Computer Vision (CV) methods to assist the visual analysis process for medical imaging data. An example of these medical imaging data is the Computed Tomography, just like the ones on Figure 1.1. Therefore, a variety of methods can be applied to aid the diagnosis of DPLDs. Once the textures of this family of pathologies is very important in the diagnosis, it is important to analyse the behaviour of CV methods as a texture classifier.

Textures present in medical digital images are complex visual patterns with particular characteristics and can be seen as powerful discriminators [Zhang et al., 2015]. Therefore, the texture classification process is an essential step in Medical Image Analysis tasks and their applications, including content-based medical image retrieved [Wei et al., 2018], image classification [Tuzuner et al., 2020, Darapureddy et al., 2021] and image segmentation [Monemian and Rabbani, 2019].

One of the main challenges in texture classification is developing an efficient descriptor invariant to rotations, scale, and lighting variations. It is possible to accurately classify textures using a large number of approaches [Humeau-Heurtier, 2019]. They include: statistical approaches, which are methods based on the extraction of statistical features (e.g., mean, variance, energy and others) from the spatial structure of images (e.g., Gray Level Co-occurrence Matrix (GLCM), Haralick [Haralick et al., 1973], Histogram of Oriented Gradient (HOG) [Sharma and Ghosh, 2015], Local Binary Patterns (LBP) [Ojala et al., 1996], and some LBP-based variants [Fernandez et al., 2011, Nguyen et al., 2016]); spectral-based approaches, which extract information from images using the frequency domain of data (e.g., Fourier Transform, Wavelet Transform and Gabor filters [Lu et al., 2018]); graph-based approaches, which use concepts of graph theory to extract information (e.g., local graph structures [Abusham and Bashir, 2011], graph of touristic walk approach [Gonçalves et al., 2016], shortest path in graphs [de Mesquita Sa et al., 2013]); deep-learning-based approaches, that use convolutional neural networks to generate filters that highlight features through a learning process (e.g., WaveletCNN [Fujieda et al., 2018] and textural convolutional neural network [Andrearczyk and Whelan, 2016]), among others.

Using graph-based approaches to extract and classify textural features of images includes converting textural images into graphs, followed by processing and extracting relevant metrics. The conversion is done by, initially, creating the set of vertices of the graph. Some works have used the method of converting the pixel directly into a vertex [de Mesquita Sa et al., 2013, Gonçalves et al., 2016]. Others have used concepts of *superpixels* and each vertex represent a region with similar characteristics [Avelar et al., 2020, Zhao et al., 2018]. The edges that connect the vertices are a representation of the relationship between the pixels/superpixels in the origi-

nal image. Common approaches have used the spatial distance and intensity to set an edge [de Mesquita Sa et al., 2013, Gonçalves et al., 2016]. Others have used superpixels as nodes and connect them based on spatial adjacency [Avelar et al., 2020, Zhao et al., 2018]. For texture recognition problems, node attributes might include pixel/region intensity, pixel spatial position, color information, neighborhood information, etc. Once the graph that represents the textural image is defined, the extraction of features can be done with some Complex Networks knowledge.

In many applications, it is possible to interpret graphs as Complex Networks (CNs). CNs are irregular and comprehensive structures inspired by empirical analysis of real networks that allows the understanding of various real systems [Cabral et al., 2014]. These systems are called "complex system" because it is impossible to predict their collective behavior from their single components. However, understanding the topological description of these systems makes predicting and controlling them possible [da Mata, 2020]. CNs are described by several metrics, which represent their topological properties. These metrics can be used as descriptors of textural images in the problem of pattern recognition. Studies using the extraction of these attributes and metrics to classify texture in images have been explored.

1.1 Proposal

We can divide the contributions of this work into two parts:

1. Several approaches of image texture classification are reviewed, implemented and analysed for classifying CT images that show DPLDs texture patterns, particularly pulmonary consolidation, and ground-glass opacity abnormalities, which are standard features of COVID-19 patients [He et al., 2020];

2. After this prior analysis, a new method is presented, based on statistical descriptors and graphs-based ones (CNs). In this approach, the images are mapped into a set of scaled graphs (CNs), and the closeness centrality metric from these graphs combined with Haralick descriptors [Haralick et al., 1973] and Local Binary Patterns (LBP) [Ojala et al., 1996] form up the texture features conducive to the classification of observed texture patterns.

1.2 Structure

This document is structured as follows. All the theoretical background for the methods analysed are described in Chapter 2. Given that we analyze a large number of classification methods, this chapter is extensive because it presents the theoretical basis for all the analyzed methods. Some related works that use concepts of textural classification in

medical images are explored in Chapter 3. The chapter starts with more general context and, later, the works cited are more focused in computed tomography of the lungs.

In Chapter 4, the empirical behaviour of the methods and how those concepts are used to characterize the images on the dataset are explained in detail. Furthermore, the proposed method is described and the whole process of mapping images to CNs, extracting features and classifying the images is explained. Chapter 5 discusses the experiments and results and compares the performance of all the methods using evaluating metrics. Finally, Chapter 6 concludes and presents suggestions for further investigation.

Chapter 2

Theoretical Background

This chapter goes through all the concepts behind the methods analysed in this work. All the theory that gives enough knowledge to understand the methods analyzed. It is an overview of the variety of methods used in the context of texture classification, like methods that use basic statistics, methods that extract measures from the frequency domain of images, methods that use concepts of graphs and Complex Networks, and knowledge of Deep Learning using Convolution Neural Network.

2.1 Statistical Methods

Statistical methods use the spatial distribution of the grey-level values within the pixel's neighborhood to extract metrics that can describe and characterize the input texture image. Furthermore, these statistical methods can be categorized based on the number of pixels used to define the feature. Methods that use one pixel to define the feature are called first-order. If two pixels are used, they are called second-order, and if three or more pixels are used, they are called higher-order statistics.

Even though the first-order methods only consider the pixel itself and not the relations with the nearby elements, they are very frequently used because, since they are histogram-based, they can extract valuable global features and they are very simple and low-cost to execute. Some of those feature are: mean, variance, maximum, minimum and kurtosis.

The second-order statistical methods use the information between two pixels and describe the interrelationship between the levels of grey in the image. These methods extract features from matrices that represent the frequency at which every grey-level value interacts with each other in a given space of the image. The Grey-Level Co-occurrence Matrix (GLCM) is one of the most used method to generate those matrices. The GLCM generate matrices by computing the frequency that every combination of different grey-level value occurs in a given direction and distance. The Haralick method uses the GLCM of the image to extract valuable metrics from it.

If the statistical method uses more than two pixels to extract information from an

image, it extracts higher-order features and can deal better with the occurrence of noise. The Local Binary Pattern (LBP) is one of those methods and it focuses on the intensity transitions within a subregion of the region of interest. LBP is a frequently used method because it performs well on texture analysis once it combines the analysis of occurrences and local structures of the image. Besides giving a good performance, it is susceptible to rotation and noise. Therefore, different variations of LBP were proposed to address some of these issues. Another well known method is the Histogram of Oriented Gradients (HOG), an effective texture descriptor commonly used for object detection and face recognition. This method's main idea is to categorize the image using the intensity distribution of the gradients or edge directions. Since it counts the occurrences of gradient orientation in local cells, it is invariant to geometric and photometric transformations.

2.1.1 Haralick Features

Proposed by Robert Haralick in [Haralick et al., 1973], these features are a set of easily computable statistical measures operated over the GLCM of the image and are capable of characterizing the texture of an image. Given an image I with size $M \times K$ and grey-levels in the range $[1, N]$, GLCM is a matrix $P \in \mathbb{N}^{N \times N}$ where each entry $\mathbf{P}(i, j)$ indicates the number of times the grey-level i occurred in a given radius and direction of a given grey-level j.

$$\mathbf{P}(i,j) = \sum_{m=1}^{M} \sum_{k=1}^{K} \begin{cases} 1, & \text{if } I(m,k) = i \text{ and } I\left(m + d_x, k + d_y\right) = j \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

Where $\delta = (d_x, d_y)$ is the displacement vector capable of varying the GLCM based on different directions and distances of the given i and j grey-level values. It is possible to generate different GLCMs with different displacement vectors. The next step is to generate the normalized matrix $p \in \mathbb{Q}^{N \times N}$ using Equation 2.2.

$$p = \frac{\mathbf{P}}{\sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{P}\left(i,j\right)} \tag{2.2}$$

With the matrix p, the Haralick features can be extracted, given that:

- $p_x(i) = \sum_{j=1}^{N} p(i,j)$ is the sum over the rows of p;
- $p_y(j) = \sum_{i=1}^{N} p(i,j)$ is the sum over the columns of p;
- $p_{x+y}(k) = \sum_{i=1}^{N} \sum_{j=1_{i+j=k}}^{N} p(i,j)$ is the sum of values of p where each intensity level i and intensity level j sum up to k for $k = 2, 3, \ldots, 2N$;
- $p_{x-y}(k) = \sum_{i=1}^{N} \sum_{j=1_{|i-j|=k}}^{N} p(i,j)$, is the sum of values of p where the absolute difference of intensity level i and intensity level j is equal to k for $k = 0, 1, \ldots, N-1$;

- $\mu_x = \sum_{i=1}^{N} i \cdot p_x(i)$ and $\sigma_x^2 = \sum_{i=1}^{N} (i-\mu_x)^2 \cdot p_x(i)$ are the mean and standard deviation of p_x;
- $\mu_y = \sum_{j=1}^{N} j \cdot p_y(j)$ and $\sigma_y^2 = \sum_{j=1}^{N} (j-\mu_y)^2 \cdot p_y(j)$ are the mean and standard deviation of p_y;
- $HX = -\sum_{i=1}^{N} p_x(i) \cdot \log p_x(i)$ is entropy of p_x and $HY = -\sum_{i=1}^{N} p_y(i) \cdot \log p_y(i)$ is entropy of p_y;
- $HXY = -\sum_{i=1}^{N}\sum_{j=1}^{N} p(i,j) \cdot \log p(i,j)$ is joint entropy of p;
- $HXY1 = -\sum_{i=1}^{N}\sum_{j=1}^{N} p(i,j) \log\{p_x(i)p_y(j)\}$ and $HXY2 = -\sum_{i=1}^{N}\sum_{j=1}^{N} p_x(i)p_y(j) \log\{p_x(i)p_y(j)\}$ are variations of the joint entropy.

Therefore, the Haralick Textural Features are:

1. Angular Second Moment (Energy): $f_1 = \sum_{i=1}^{N}\sum_{j=1}^{N} \{p(i,j)\}^2$.
2. Contrast: $f_2 = \sum_{i=1}^{N}\sum_{j=1}^{N} (i-j)^2 p(i,j)$
3. Correlation: $f_3 = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} (ij)p(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y}$
4. Sum of Squares (Variance): $f_4 = \sum_{i=1}^{N}\sum_{j=1}^{N} (i-\mu)^2 p(i,j)$.
5. Inverse Difference Moment: $f_5 = \sum_{i=1}^{N}\sum_{j=1}^{N} \frac{1}{1+(i-j)^2} p(i,j)$.
6. Sum Average: $f_6 = \sum_{i=2}^{2N} i p_{x+y}(i)$.
7. Sum Variance: $f_7 = \sum_{i=2}^{2N} (i-f_8)^2 p_{x+y}(i)$.
8. Sum Entropy: $f_8 = -\sum_{i=2}^{2N} p_{x+y}(i) \log\{p_{x+y}(i)\}$.
9. Entropy: $f_9 = -\sum_{i=1}^{N}\sum_{j=1}^{N} p(i,j) \log(p(i,j))$.
10. Difference Variance: $f_{10} =$ variance of p_{x-y}.
11. Difference Entropy: $f_{11} = -\sum_{i=0}^{N-1} p_{x-y}(i) \log\{p_{x-y}(i)\}$.
12. Information Measures of Correlation 1: $f_{12} = \frac{HXY - HXY1}{\max\{HX,HY\}}$
13. Information Measures of Correlation 2: $f_{13} = (1 - \exp[-2.0(HXY2 - HXY)])^{1/2}$
14. Maximal Correlation Coefficient: $f_{14} =$ (Second largest eigenvalue of $Q)^{1/2}$, where $Q(i,j) = \sum_k \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}$.

2.1.2 Local Binary Pattern

Initially proposed in [Ojala et al., 1996], Local Binary Pattern (LBP) is a method that uses the grey-levels of a central pixel and its neighborhood to classify this central pixel and, from it, generate a new image used to characterize the texture. However, this initial approach was susceptible to noise and rotation. Therefore, [Ojala et al., 2002] proposed an improved version of the method.

To define a texture T in a small surrounding area of a monochromatic texture image, it is used a distribution of the grey levels of P image pixels, where $P > 1$:

$$T = t(g_c, g_0, \ldots, g_{P-1}), \tag{2.3}$$

where g_c is the gray level value of the central pixel in the local neighborhood and g_p for $p = 0, 1, \ldots, P-1$ correspond to the gray level of the P equally spaced pixels on a circle with radius R around the central pixel, where $R > 0$. In Figure 2.1, the neighborhood of elements can be seen for different values of P and R.

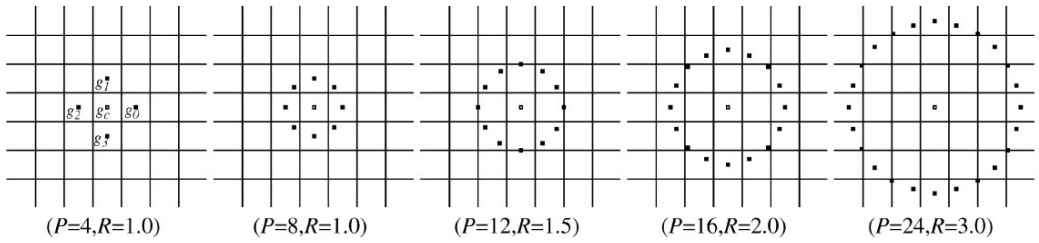


Figure 2.1: Circularly symmetric neighbor sets for different (P, R).
Source: [Ojala et al., 2002]

To achieve gray-scale invariance, the g_c value is subtracted from the gray-level value of the pixels from the circle, giving:

$$T = t(g_c, g_0 - g_c, g_1 - g_c, \ldots, g_{P-1} - g_c). \tag{2.4}$$

Assuming that the differences $g_p - g_c$ are independent of g_c,

$$T \approx t(g_c)t(g_0 - g_c, g_1 - g_c, \ldots, g_{P-1} - g_c). \tag{2.5}$$

Since the distribution $t(g_c)$ describes the overall luminance of the image, which is unrelated to local image texture, it does not provide useful information for the analysis. Hence, distribution T is defined as, approximately:

$$T \approx t(g_0 - g_c, g_1 - g_c, \ldots, g_{P-1} - g_c). \tag{2.6}$$

This texture operator is highly discriminative once it gives information of the occur-

rence of differents textures and various patterns. For exemple, a constant neighborhood gives all zeros. If changes in luminance occur, the differences $g_p - g_c$ are not affected. To achieve invariance with respect to the scaling of the gray scale, it is considered only the signs of the difference and not the difference itself:

$$T \approx t(s(g_0 - g_c), s(g_1 - g_c), \ldots, s(g_{P-1} - g_c)), \tag{2.7}$$

where

$$s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.8}$$

A binomial factor 2^p is assigned for each sign $s(g_p - g_c)$ to generate a single value $LBP_{P,R}$ to categorize the texture on this neighborhood:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p. \tag{2.9}$$

The $LBP_{P,R}$ operator produces 2^P different output values which are the possible values of the local binary pattern in the neighborhood. If the image is rotated, the g_p values are gonna change because the g_0 is always going to be the first element on the right of the central pixel. Therefore the current $LBP_{P,R}$ operator is going to change when a rotation occur on the same neighborhood. To remove the rotation variability a new operator is defined:

$$LBP^{ri}_{P,R} = \min\{ROR(LBP_{P,R}, i) \quad | \quad i = 0, 1, \ldots, P-1\}, \tag{2.10}$$

where $ROR(LBP_{P,R}, i)$ performs a circular bit-wise right shift on the P-bit number $LBP_{P,R}$ by i bits. The min operator sets that this operator will choose the minimum $LBP_{P,R}$ for the neighborhood.

To improve the operator, it is introduced the concept of "Uniform" patterns. Uniform circular structures present very few spatial transitions. Empirically, was proved that these uniform patterns are fundamental properties of texture and very important in the process of characterizing these structures. To formally define these Uniform patterns, it is introduced an uniformity measure U, which corresponds to the number of spatial transitions (bitwise 0/1 changes) in the pattern. The new operator that takes this concept in consideration is defined as:

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & \text{if } U(LBP_{P,R}) \leq 2 \\ P + 1 & \text{otherwise,} \end{cases} \tag{2.11}$$

where

$$U(LBP_{P.R}) = \mid s(g_{P-1} - g_c) - s(g_0 - g_c) \mid + \sum_{p=1}^{P-1} \mid s(g_p - g_c) - s(g_{p-1} - g_c) \mid . \quad (2.12)$$

So, in other words, this operator will count the number of spatial transitions in the neighborhood. If it is lesser or equal to 2, $LBP_{P.R}$ will be the number of bits 1, otherwise it will be the number of elements in the neighborhood.

2.1.3 Histogram of Oriented Gradients (HOG)

The central idea of this method [Dalal and Triggs, 2005] is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid. In the context of texture analysis and recognition, the HOG descriptors will characterize the texture using the distribution of local intensity gradients or edge directions. The method has presented great performance in problems of object detection and face recognition.

Initially, the image is divided in smaller spatial regions called *cells*. For each cell, a one dimensional histogram of gradient directions or edge orientations is generated over the cells' pixels. So, for each cell of the image, a 1-D vector is generated. The entire image is then characterized by the combination of all the vectors from the cells. To better the performance and to better the invariance to different illuminations settings, a normalization is operated. To achieve that, the original image is divided into other spatial regions called blocks that are formed by a group of cells. The local histogram energy of these blocks are measured and accumulated to normalize the result of all the cells inside the block. The normalized descriptor blocks are referred as the Histogram of Oriented Gradients (HOG) descriptors. The grid described can overlap, therefore the pixels can be part of more than one block or cell.

2.2 Spectral Methods

Spectral methods are a class of methods used to represent the textures in multiple resolutions and scales. They can analyse the texture's frequency content either exclusively in spatial domain (using Laws' Filters) or exclusively in the frequency domain (using methods like Fourier Transform). Alternatively, these methods can also be used to analyze both the frequency and spatial domains of the texture, which can be achieved through techniques like Gabor and Wavelet Transform.

2.2.1 Laws' Texture Energy

Introduced in [Laws, 1980], this method was one of the pioneers on the texture characterization field using the spectral information of the image. Besides being rotationally invariant, it presents great performance on characterizing textures. This method consists of convolving the image with a set of filters and extracting statistics from the filter response. The set of filters of the method is based on the 3 simple vectors of length 3: center weighted local averaging $L_3 = (1, 2, 1)$, edge detection $E_3 = (-1, 0, 1)$ and spot detection $S_3 = (-1, 2, -1)$.

If these vectors are convolved with themselves we have a set of filter with length 5: center weighted local averaging $L_5 = (1, 4, 6, 4, 1)$, edge detection $E_5 = (-1, -2, 0, 2, 1)$ and spot detection $S_5 = (-1, 0, 2, 0, -1)$. Furthermore, it is possible to obtain the vectors with length 7: $L_7 = (1, 6, 15, 20, 15, 6, 1)$, $E_7 = (-1, -4, -5, 0, 5, 4, 1)$ and $S_7 = (-1, -2, 1, 4, 1, 1, -2, -1)$. Given the family of vectors of length l, the Laws filters of size $l \times l$ can be obtained by multiplying different combination of those one dimensional vectors. For example, for the family of vectors of length 5, we have the 5×5 filters:

$$\begin{aligned} LL &= L_5^T L_5 \\ EE &= E_5^T E_5 \\ SS &= S_5^T S_5 \\ LE &= L_5^T E_5 \\ LS &= L_5^T S_5 \\ EL &= E_5^T L_5 \\ ES &= E_5^T S_5 \\ SL &= S_5^T L_5 \\ SE &= S_5^T E_5 \end{aligned} \tag{2.13}$$

To extract information from the image and describe the texture, these filters are convoluted over the image and statistical measures (e.g., energy) [Giakoumoglou, 2021] are extracted from the filter responses. These filter responses can be viewed as two dimensional signals, and, therefore, signal processing concepts can be used to extract information. The measure of energy represents a numerical value of total magnitude of the signal/image. Some measures are:

- Texture energy from the LL filter;
- Texture energy from the EE filter;
- Texture energy from the SS filter;
- Average texture energy from LE and EL filters: $(LE + EL)/2$;
- Average texture energy from ES and SE filters: $(ES + SE)/2$;

- Average texture energy from LS and SL filters: $(LS + SL)/2$;

These measures will be used to characterize the input images.

2.2.2 Gabor Filters

Being very effective in texture representation and differentiation, the Gabor filters are a family of filters generated by the modulation of a Gaussian kernel using a sinusoidal plane wave. The method can deal very well with noise and changes in illumination, but fail to achieve the same performance with variations in rotation and scale. This bank of 2-D filters is formed by Gabor filters generated at different scale and orientations [Lu et al., 2018]. Let $I(x, y)$ be an input image and $g_{\lambda,\theta}(x, y, \varphi, \sigma, \gamma)$ be a Gabor kernel function with scale value λ and orientation value θ. The filter response is defined as the convolution of $I(x, y)$ and $g_{\lambda,\theta}(x, y, \varphi, \sigma, \gamma)$:

$$G_{\lambda,\theta}(x, y) = I(x, y) * g_{\lambda,\theta}(x, y, \varphi, \sigma, \gamma), \tag{2.14}$$

where

$$g_{\lambda,\theta}(x, y, \varphi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi\frac{x'}{\lambda} + \varphi\right), \tag{2.15}$$

$$x' = x\cos\theta + y\sin,\theta, \tag{2.16}$$

$$y' = -x\sin\theta + y\cos\theta, \tag{2.17}$$

$\gamma = 0.5$ is a constant, λ represents the wavelength (so $1/\lambda$ is spatial frequency) and φ correspond to anti-symmetric functions ($\varphi = -\pi/2$ or $\varphi = \pi/2$). The ratio σ/λ is given by

$$\frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{\ln 2}{2}}\frac{2^b + 1}{2^b - 1}, \tag{2.18}$$

where b is the half response spatial frequency bandwidth.

With this definition, the amount of filters responses is going to depend on the chosen n frequencies $1/\lambda$ and m orientations θ, giving $n \cdot m$ responses. Figure 2.2 shows the 24 filters generated by 6 values of θ and 4 values of $1/\lambda$.

After that, statistical information can be measured on the filter responses $G_{\lambda,\theta}(x, y)$. These responses can be viewed as two-dimensional signals and we can use concepts of signal processing to extract metrics. One very commonly used metric is the Gabor energy, which, in this context, is a way to measure the total magnitude of the image/signal. For each response $G_{\lambda,\theta}(x, y)$ generated, a value of energy can be extracted using

$$E_{\lambda,\theta}(G) = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\left|G_{\lambda,\theta}(x, y)\right|^2, \tag{2.19}$$

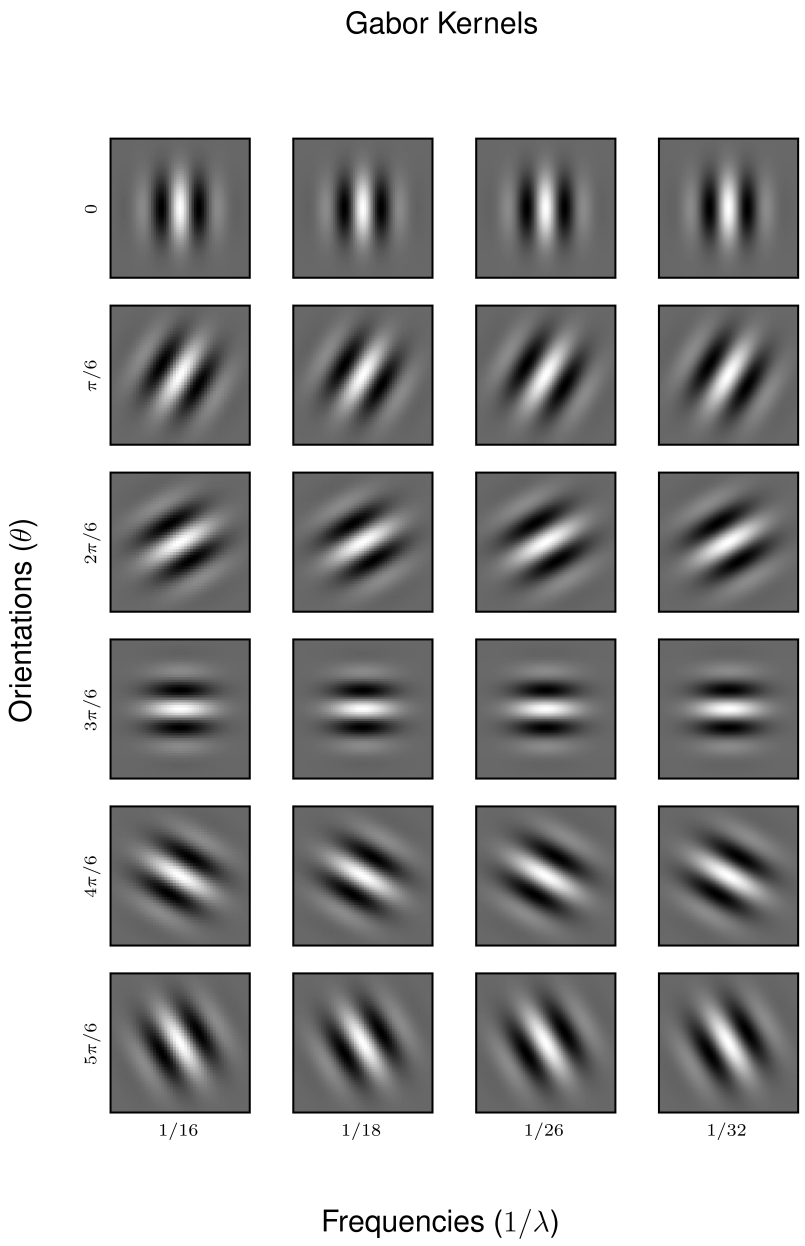


Figure 2.2: Examples of Gabor Kernels for different values of orientations θ and frequencies $1/\lambda$. In this, θ = (0, $\pi/6$, $2\pi/6$, $3\pi/6$, $4\pi/6$, $5\pi/6$) and $1/\lambda$ = (1/16, 1/18, 1/26, 1/32).

where $|G_{\lambda,\theta}(x,y)|^2$ is the magnitude of the response $G_{\lambda,\theta}(x,y)$. This information will be used to characterize the input image.

2.2.3 Fourier Transform

The Fourier transform is a fundamental approach on the frequency domain analysis [Ghalati et al., 2022] once it is capable of decomposing a given input and convert its information from time or spatial domain to the frequency domain using mathematical fundamentals. It is based on the concept that every signal (one dimension) can be decomposed into a sum of sine and cosine waves with different amplitudes and frequencies. These sine and cosine waves are called the frequency components of the original signal. This concept can be extended to images (two dimension) and used to decompose images into their frequency components represented by a combination of vertical and horizontal sine and cosine waves of different frequencies. Given an image f of size $M \times N$, the Fourier Transform F of the image is

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{N-1} \sum_{v=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+vy/N)}, \tag{2.20}$$

where u, v are the coordinate from the frequency domain. Alternatively, it is possible to convert the frequency domain image back to the spatial domain using the Inverse Fourier Transform defined as

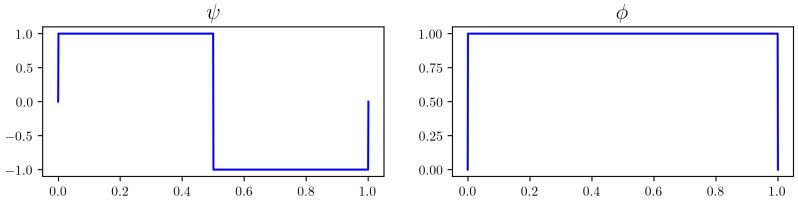
$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux/M+vy/N)}. \tag{2.21}$$

These operations give the capability to manipulate and perform operations on the image in the frequency domain and return it to the original domain. One example of usage is filtering out the noise of the image without losing important information in the original domain. Furthermore, it is possible to work with the frequency domain image F to characterize the texture on images. With the frequency domain image, it is possible to extract spectral information of the original input using some statistical measures (e.g., energy).

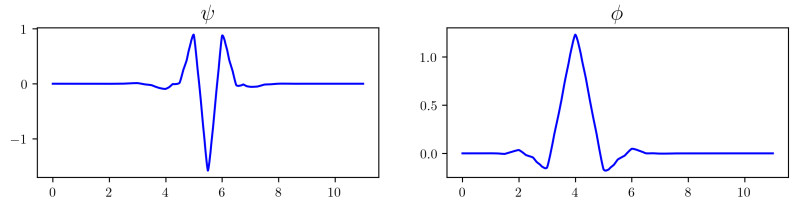
2.2.4 Wavelet Transform

The Wavelet Transform was proposed as an alternative to the Fourier Transform once the latter does not maintain time information in the frequency domain. Fourier Transform uses the input signal to plot the contribution of different sine waves at different frequencies but can't inform at what time those frequencies begin or end. The Wavelet Transform comes as an alternative for that problem and is used as method of Multiresolution Analysis of data.

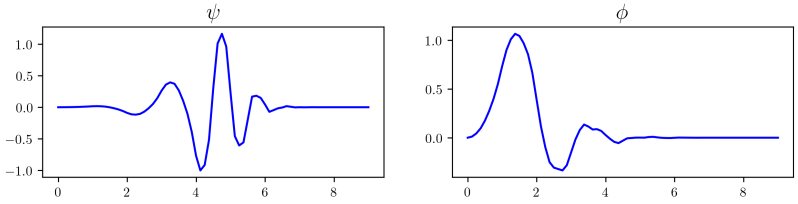
Unlike Fourier Transform that use sine waves to represent the signal, the method uses analysing functions called wavelets, which can be seen as sine waves restrained in time. Those functions need to have zero mean and the area under the curve has to be finite, which is what makes the function localized in time.



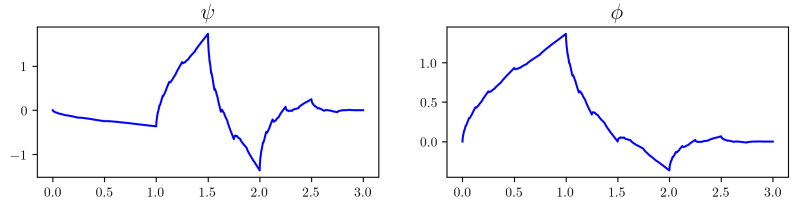
(a) Haar Wavelet



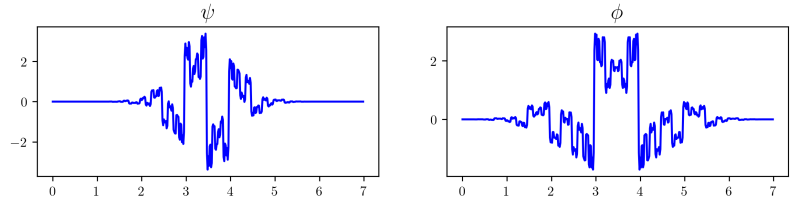
(b) Coiflets Wavelet



(c) Daubechies Wavelet



(d) Symlets Wavelet



(e) Biorthogonal Wavelet

Figure 2.3: Examples of some families of wavelet commonly used. ψ is the wavelet function and ϕ is the scaling function.

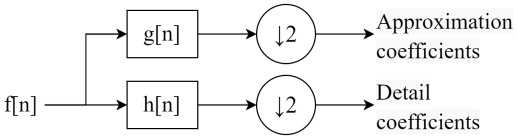


Figure 2.4: Decomposition of signal x into detail and approximation coefficients using high-pass (h) and low-pass (g) filters, respectively. The coefficients are generated after the downsample step.

Wavelets are used in families and each of the families have the mother wavelet (commonly called the wavelet function ψ) and the wavelet father (commonly called the scaling function φ). The wavelet function ψ represents the high frequency components of the input and the scaling function φ represent the low frequency components of the input. Some families of wavelets can be seen on Figure 2.3.

Wavelet functions can be translated by a value k and scaled by a value j. Therefore, $\psi_{j,k} = \psi(\frac{t-k}{j})$ is a daughter wavelet function derived by the wavelet mother ψ. The same can be define for daughter scaling functions $\varphi_{j,k}$. This manipulation will allow to determine the influence of a given wavelet at a certain scale and time. Therefore, by varying the scale and translation parameters, it is possible to determine what frequencies are prominent in the signal at a given time point.

The wavelet series expansion of a function $f(x)$ with respect to wavelet function $\psi(x)$ and scaling function $\varphi(x)$ can be represented as

$$f(x) = \sum_k c_{j_0}(k)\varphi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_k d_j(k)\,\psi_{j,k}(x)\,, \tag{2.22}$$

where c_{j_0} id the *approximation coefficient* and d_j for $j \geq j_0$ are the *detail coefficients*.

If the function $f(x)$ being expanded is discrete, the approximation and detail coefficients of the expansion are its Discrete Wavelet Transform (DWT) and the expansion itself in Equation 2.22 is the function's Inverse Discrete Wavelet Transform (IDWT).

Once digital (discrete) signals are the focus, the implementation of the DWT can be achieved using digital filters and downsampling methods, once it will generate details of the data at different scales.

The digital signal is passed through a low-pass filter g, producing the approximation coefficient, and through a high-pass filter h, producing the detail coefficient. Since half the frequencies were removed, according to Nyquist's theorem, the resulting signal can be downsampled by 2. This can be seen in Figure 2.4.

The signal can be decomposed in more than one level. For this, the process is repeated and the approximation coefficient at level j is decomposed into the approximation and detail coefficients of level $j+1$ using low- and high-pass filter with half the cut-off frequency of level j. The Figure 2.5 illustrates this with 3 decomposition levels.

This transformation can be easily converted to deal with 2-D information. The process

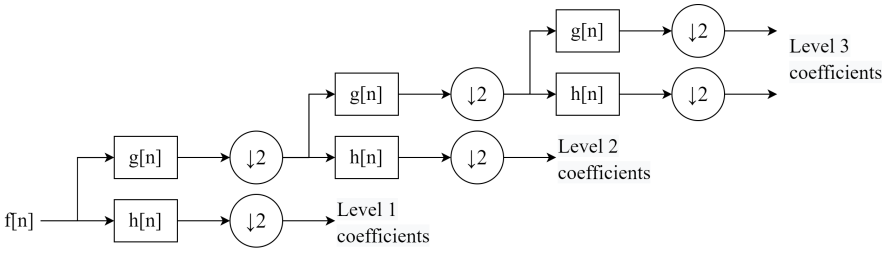


Figure 2.5: Decomposition of signal x into detail and approximation coefficients with 3 levels.

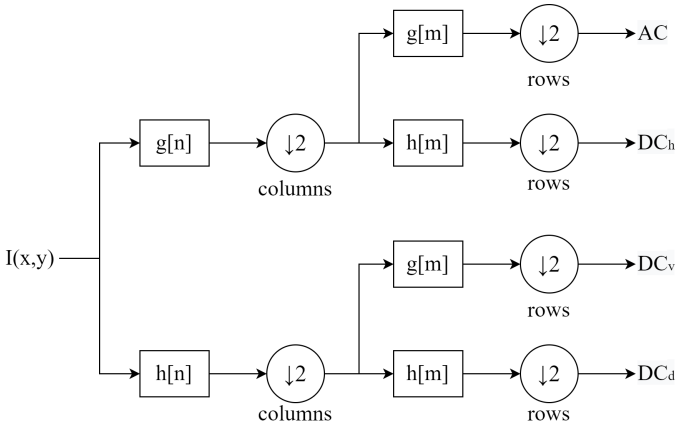


Figure 2.6: Decomposition of an image I into the approximation coefficient (AC) and detail coefficients (DC_v, DC_h and DC_d)

can be seen on Figure 2.6. Initially, the rows go under the low-pass (g) and high-pass (h) filters and the number of columns are downsampled. Then, each column goes under the filters and the number of rows is downsampled. The combination of each operation generates 4 sub-images: AC (Approximation Coefficient) is the resultant image where the low-pass filter went on both directions; DC_d (Detail Coefficient - Diagonal) is the resultant image of the high-pass filter on both directions; low-pass on each row and high-pass on each column generated DC_h (Horizontal); and high-pass on each row and low-pass on each column generated DC_v (Vertical).

The Figure 2.7 shows a real example of how the image and sub-images are displayed. On the left, there is the original image $I(x, y) = AC_j$. On the right, there are the 4 sub-images generated: $AC_{(j-1)}$, on the top-left; $DC_{h,(j-1)}$ on the top-right; $DC_{v,(j-1)}$ on the bottom-left; and $DC_{d,(j-1)}$ on the bottom-right.

Furthermore, there is a variation of the DWT called Stationary Wavelet Transform (SWT) which does not perform the downsampling of the rows and columns, but performs an upsampling of the filters instead. This variations provides shift invariance to the transformation but it has a higher computation cost.

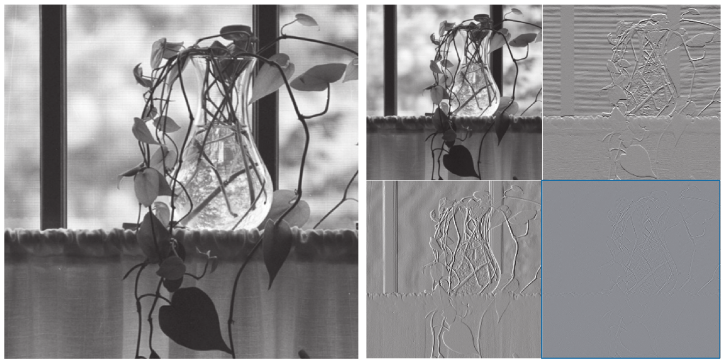


Figure 2.7: Usual arrangement of the original image and the subimages generated by the 2-D DWT. Source: [Gonzalez and Woods, 2018]

2.2.5 Fractal Dimension

The fractal dimension is often used in mathematics to describe the complexity of fractal patterns. The complexity is quantified using a ratio of the change in detail to the change in scale. It was first applied to characterize complicated shapes where the details seemed more important than the gross pictures [Albers and Alexanderson, 2008]. For ordinary shapes, the fractal dimension is the same as the topological dimension: 0-dimension for points, 1-dimension for lines, 2-dimension for planes, 3-dimension for volumes. But if the fractal dimension exceeds the topological dimension, it is considered to have a fractal geometry.

The Fractal Dimension D_f [Wu et al., 1992] of an image $I(x, y)$ can be can be estimated using

$$E(\Delta I^2) = c(\Delta r)^{(6-2D_f)}, \tag{2.23}$$

where $E(\cdot)$ is the expectation operator, ΔI is the intensity variation $I(x_2, y_2) - I(x_1, y_1)$, c is a constant and Δr is the spatial distance $(x_2, y_2) - (x_1, y_1)$. A simpler method is to estimate the H parameter from the relationship

$$E(|\Delta I|) = k(\Delta r)^H, \tag{2.24}$$

where $k = E(|\Delta I|)_{\Delta r=1}$. From 2.23 and 2.24, it is possible to simplify the process of finding the fractal dimension D_f by using

$$D_f = 3 - H. \tag{2.25}$$

A large value of D_f indicate a small value of H and represents rough surfaces. Small values of D_f indicates large value of H and represents smoother surfaces. Given that image I has size $M \times M$, the intensity difference vector is defined as IDV $= [id(1), id(2), \ldots, id(s)]$, where s is the maximum possible distance between pixels on

the image and $id(k)$ is the average of the absolute intensity difference between all pairs of pixels with horizontal or vertical distance k. $id(k)$ is defined as

$$id(k) = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{M-k-1}|I(x,y) - I(x,y+k)| + \sum_{x=0}^{M-k-1}\sum_{y=0}^{M-1}|I(x,y) - I(x+k,y)|}{2M(M-k-1)}. \tag{2.26}$$

The value of H can be obtained by using least-squares linear regression to estimate the slope of the curve of $id(k)$ versus k in *log-log* scales. H is known as the *Hurst Coefficient* or *Hurst Exponent* and it characterizes a curve using statistical measure of the curve's behaviour and estimation using *Power Law*. Besides the Fractal Dimension, this concept is often used in Time Series field to measure the randomness of a data series.

Therefore, with those definitions, a feature vector [Wu et al., 1992] that uses multiresolution information can be defined as

$$MF = \left(H^{(m)}, H^{(m-1)}, \ldots, H^{(m-n+1)}\right), \tag{2.27}$$

where $M = 2^m$ is the size of the image, $H^{(p)}$ is the H parameter estimated from image I at resolution p and n is the number of resolution levels chosen. So, $H^{(m)}$ will be the Hurst Coefficient of the image with original resolution, $H^{(m-1)}$ will be the Hurst Coefficient of the image with half the resolution and so on. The Multiresolution Fractal (MF) feature vector has information of the roughness, lacunarity and regularity of the image.

2.3 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is the process of finding the principal components of a set of data and then using those components to perform a change of basis in the data. Usually, it is used as a way to reduce dimensionality once those principal components can be used to represent the data in the correspondent dimension and, therefore, can remove the dimension where the principal components has least variation.

Mathematically, the PCA is an orthogonal linear transformation that changes the data coordinate system to one where the first coordinate is the principal component with greatest variation. The second coordinate is the second greatest variance and so on [Jollife, 2010].

Let the data be a matrix $\mathbf{X}$ of size $n \times p$ where each row is an occurrence of the experiment and each columns is a feature. The transformation is a set of size l formed by p-dimensional vectors of weights $\mathbf{w}_{(k)} = (w_1, \ldots, w_p)_{(k)}$ that map each row $\mathbf{x_{(i)}}$ of $\mathbf{X}$ to a new vector of principal component scores $\mathbf{t}_{(i)} = (t_1, \ldots, t_l)_{(i)}$, given by $t_{k(i)} = \mathbf{x_{(i)}} \cdot \mathbf{w_{(k)}}$ for $i = 1, \ldots, n$ and $k = 1, \ldots, l$ in a way that the variables $t_1, \ldots, t_l$ of $\mathbf{t}$ considered over the data inherits as much variance from the data as possible while still being orthogonal to

the previous components. Usually, the number of principal components t is smaller than p to help reduce the dimensionality of the data.

To maximize the variance of the data, the first weight vector $\mathbf{w_{(1)}}$ satisfies

$$\mathbf{w}_{(1)} = \arg\max_{|\mathbf{w}|=1} \left\{ \sum_i (t_1)^2_{(i)} \right\} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w} \right)^2 \right\}. \tag{2.28}$$

Since $\mathbf{w_{(1)}}$ is chosen to be a unit vector, this can be written in the matrix form as

$$\mathbf{w}_{(1)} = \arg\max \left\{ \frac{\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w}}{\mathbf{w}^\mathsf{T}\mathbf{w}} \right\}. \tag{2.29}$$

With $\mathbf{w_{(1)}}$ found, the first principal component of a data vector $\mathbf{x_{(i)}}$ can be given by $t_{1(i)} = \mathbf{x_{(i)}} \cdot \mathbf{w_{(1)}}$. Subsequently, the k-th component can be found by subtracting the first $k-1$ principal components of $\mathbf{X}$

$$\hat{\mathbf{X}}_k = \mathbf{X} - \sum_{s=1}^{k-1} \mathbf{X}\mathbf{w}_{(s)}\mathbf{w}^\mathsf{T}_{(s)} \tag{2.30}$$

and then finding the weight vector that extracts maximum variance from $\hat{\mathbf{X}}_k$:

$$\mathbf{w}_{(k)} = \arg\max \left\{ \frac{\mathbf{w}^\mathsf{T}\hat{\mathbf{X}}_k{}^\mathsf{T}\hat{\mathbf{X}}_k\mathbf{w}}{\mathbf{w}^\mathsf{T}\mathbf{w}} \right\}. \tag{2.31}$$

Therefore, the k-th principal component of data vector $\mathbf{x_{(i)}}$ can be given by $t_{k(i)} = \mathbf{x_{(i)}} \cdot \mathbf{w_{(k)}}$. The full principal component decomposition of $\mathbf{X}$ can be given by:

$$\mathbf{T} = \mathbf{X}\mathbf{W}, \tag{2.32}$$

where $\mathbf{W}$ is a $p \times p$ matrix of weights whose columns are the eigenvectors of $\mathbf{X}^T\mathbf{X}$.

2.4 Graph Theory and the Closeness Centrality Metric

Graphs Theory is a field that studies the relations between entities of a given set. A set of these relations can be represented in the form of *graphs* formed by the *vertices* (entities) and *edges* (relations between the entities). A graph G is a mathematical structure consisting of a set of vertices V and a set E of edges connecting the vertices: $G = (V, E)$, where V is a set and $E \subseteq V \times V$ is a set of ordered pairs of elements from V.

The graph is said to be *undirected* if the existence of an edge $(u, v) \in E$ implies the existence of and edge $(v, u) \in E$. Otherwise, the graph is called *directed* graph. The edges of the graph can be assigned a numerical value by a *weight* function $w : E \longrightarrow \mathbb{R}$. If the edges of a graph have weights, it is called *weighted graph*, otherwise are called *unweighted*

graph.

Graphs can be represented using a matrix of size $N \times N$, where N is the amount of vertices in the graph. Therefore, this matrix represents the existence of an edge between every pair of vertex in the graph. This matrix A is called *adjacency matrix*. For an unweighted graph, $A(u, v) = 1$ means the edge (u, v) exists in set E and $A(u, v) = 0$ indicates the opposite. For a weighted graph, the adjacency matrix can be replaced by a matrix W that each entry $W(u, v)$ indicates a numerical value representing the weight of the edge between the vertices u and v. If the edge does not exist, it returns 0.

An important property of the vertices is the degree or valency k. The degree k_u are the amount of edges that connect to the node u, where self connecting edges ($(u, u) \in E$) count as 2. For directed graphs, the degree can be separated as the amount of edges that has the node u as the terminal node ${k_u}^{in}$ (in-degree) and the amount of edges that has u as the initial node ${k_u}^{out}$ (out-degree). In this case, $k_u = {k_u}^{in} + {k_u}^{out}$.

In graph theory, a very common object of study is the problem of the *shortest paths*. This problem consists of finding the shortest path between two vertices using the edges information. Therefore, the shortest path between two vertices is a series of adjacent nodes that present the lowest total weight. It can also be thought as the shortest distance or cheapest cost between two points. Various algorithms were developed to solve this problem and one of those algorithms is the *Dijkstra's algorithm.*

It consists of separating the vertices in two sets: *visited set* and *unvisited set*. Initially the visited set is formed by a chosen initial vertex v_i and the unvisited set is formed by all the other nodes. The shortest distance from v_i to itself is 0 and to all the other nodes is set to infinity. The iteration starts by calculating the tentative distance through the current node to all other nodes. If this tentative distance is smaller than the current shortest path, the shortest path is updated. The next visited node will be the one on the unvisited set with the shortest path. This iteration is repeated until all nodes are on the visited set. Despite being only applicable to non-negative weight graphs, this algorithm is very commonly used for its simplicity and good performance.

A further study, specializes at describing the behaviour of real world phenomenons using graphs and statistical physics analysis. This area is called the Complex Networks and it is capable of describing real systems using graphs with complex topological structure. In this field, in order to extract information and try to describe the behaviour of those networks, multiple metrics were proposed.

One of the metrics is the *Closeness Centrality*. This metric gives a global information of the graph or network and, for each node, it is given a numerical value that indicates how central is the node compared to the other nodes of the network. For a connected graph G, the Closeness Centrality metric CC of a vertex v_i can be seen as the inverse of the sums of the minimum path distances from v_i to all other vertices of the network. However, this definition is unsuitable when the network is disjointed, as some vertices are

not reachable. Thus, a more common way to calculate the Closeness is from the sum of the inverses of the shortest path distances

$$CC(v_i) = \frac{1}{N-1} \sum_{j \neq i} \frac{1}{d_{v_i,v_j}}, \tag{2.33}$$

where, N is the number of vertices in the Graph; d_{v_i,v_j} is the length of the shortest path between two vertices v_i and v_j; and, therefore, $\frac{1}{d_{v_i,v_j}} = 0$ if there are no path between v_i and v_j. With this, the more central is the node, bigger its closeness centrality measure is and better is its view over the flow of data through the network [Lü et al., 2016].

Another metric that can be noted is the diffusion, which is one of the most studied phenomena in physics and biology [Comin et al., 2012]. On physics, diffusion is the movement that particles perform from an area of high concentration to an area of low concentration until stability is reached. This concept can be used on directed graphs to describe the flow of information inside a network with complex topology. In practice, a random walker is used to generate a probability distribution of the amount of time each node is visited and how much information goes through it.

Mathematically, if the walker is at a node v_j at a time $t-1$, the probability of finding this walker at a node v_i at a time t is

$$P_i(t) = \sum_{j=1}^{N} \frac{A_{ij}}{k_j} P_j(t-1), \tag{2.34}$$

where k_j is the the degree of node j, N is the total number of nodes in the network and A_{ij} is the entry of the adjacency matrix A. After a long period of time, the system is guaranteed to reach the equilibrium [Noh and Rieger, 2004] and $P_i(t) = P_i(t-1) = P_i^{\infty}$. Empirically, for directed graphs, the activity of each node is estimated by starting random walks on each node. So, given the walker is at the node v_i, the probability of going to a node v_j is

$$p(v_i, v_j) = \frac{w_{v_i,v_j}}{\sum_{v_k \in V} w_{v_i,v_k}}. \tag{2.35}$$

Then, the activity of a node $\alpha(v_i)$ is given by the amount of times the node v_i was visited by the walker after all the random walks [Gonçalves et al., 2016].

2.5 Deep Learning

Deep Learning (DL) is a sub-area of the Machine Learning (ML) field that is responsible for performing the process of finding patterns and making decisions using an artificial neural network formed by neurons. The neurons are units that takes multiple inputs and generates only one output. These units are distributed in multiple layers and can be

configured to perform different calculations of the output. The combinations of the layers makes up the network.

The first layer is the input, the last layer is the output and the middle ℓ layers are called the hidden layers. If all the neurons are connected with the previous and next neuron layers, it is called a *fully connected network* and it can be seen on Figure 2.8.

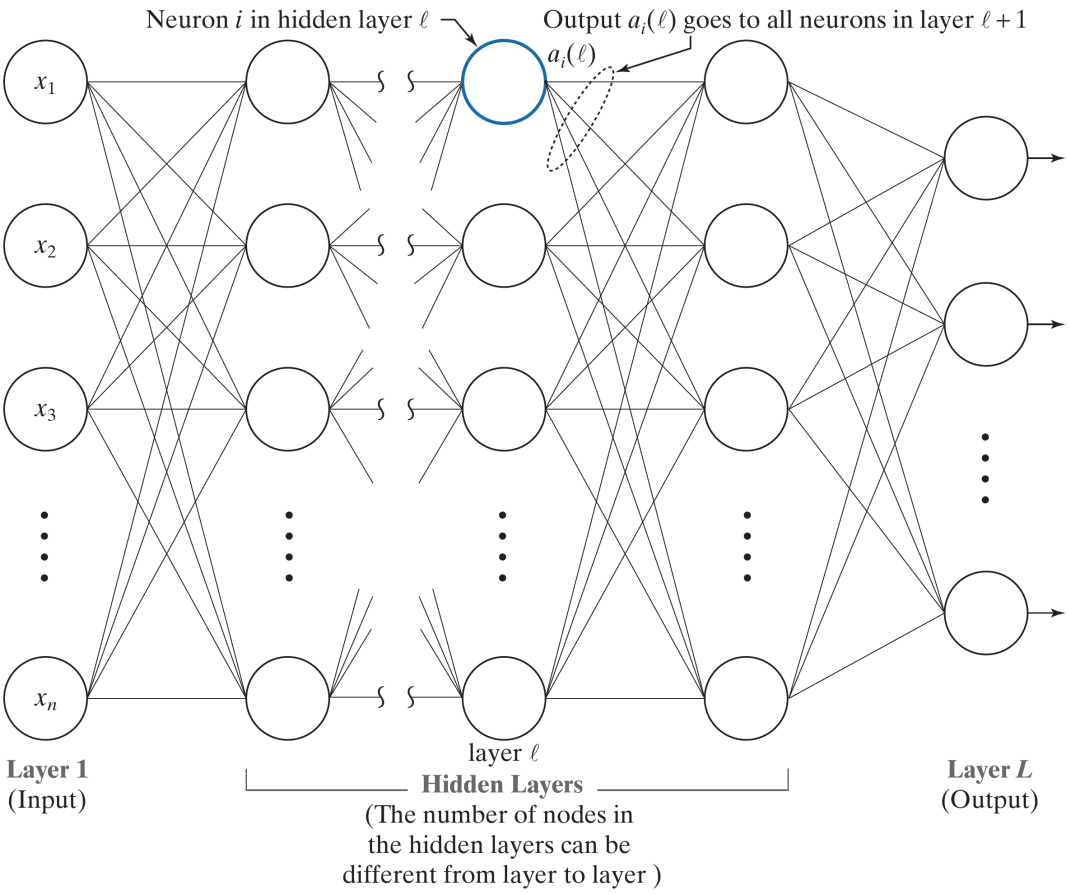


Figure 2.8: General fully connected network architecture with input of size n, ℓ hidden layers and an output layer L. Source: [Gonzalez and Woods, 2018]

The process of training a network has the objective of finding the best weights w and biases b that represent the data and gives the output value wanted. It is given by using 2 main steps: the *forward pass* and the *backpropagation*. The forward pass is the step where the data is passed through the network to the output using all the current weight and biases of each neuron.

This operation is done over all the layers and it will give an output on the last layer. This output will be compared to the actual target output and the difference will be used to adjust the weights and biases of all the neurons of the network. This step is the backpropagation. The objective of this step is to find the optimal weights and biases for the network as it will minimize an error function E.

This can be visualized as the weights and biases being altered by the amount of influence they had in the error function. Mathematically, there is

$$\begin{aligned} w_{k+1} &= w_k - \alpha \left[\frac{\partial E}{\partial w}\right]_{w=w_k}, \\ b_{k+1} &= b_k - \alpha \left[\frac{\partial E}{\partial b}\right]_{b=b_k}, \end{aligned} \tag{2.36}$$

where w_i and b_i are the weights and biases at a given step i, $\partial E/\partial w$ indicates the influence of the weight w in the error function E and α is the learning rate of the network and it is a hyperparameter given by the user.

Using this concept of deep networks, a new type was proposed to work directly with images: Deep Convolutional Neural Network. It is based on the concept that the network itself will be able to find patterns and the features of the image. The overall behaviour of Deep Neural Networks and Convolutional Neural Network (CNN) are very similar and only differs on some operations. The basic structure of a CNN can be seen in Figure 2.9.

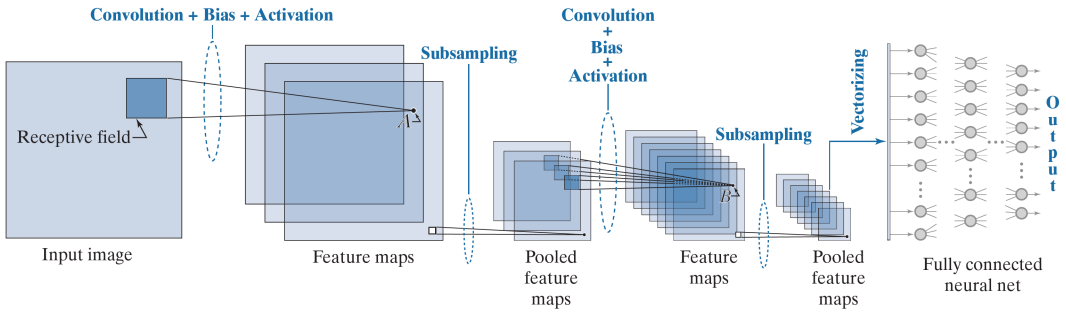


Figure 2.9: Basic structure of a Deep Convolutional Neural Network. Source: [Gonzalez and Woods, 2018]

The first operations executed is the convolution which consists of moving a receptive field over the image and generating other images called feature maps. These receptive fields are also called *kernels* and they are a small $d \times d$ matrix formed by weights w. As the kernel is passed through the image, the sum product of the weights and the pixels contained in the receptive field at that position is executed and summed with a bias. Then, this single value is passed through an *activation function* and the pixel of the feature map is generated. The *stride* is the amount of increments the kernel takes to convolve over the image. The strides are used to reduce the size of the data. For each receptive field, a feature map is generated and are referred to as *convolutional layer*.

The next step is the *subsampling* or *pooling*. It has the objective of reducing the spatial resolution of the feature map and, therefore, achieving translational invariance. This consist of dividing the feature map in small neighborhoods (usually of size 2×2, which results in 1/4 of the original size of the feature map) and generating a single value

to represent this neighborhood in the next layer. Some common pooling methods are: average pooling, take the average of the pixels in the neighborhood; max-pooling, choose the larger value between the pixels in the neighborhood; L_2, the values is given by the square root of the sum of the pixels in the neighborhood.

In Figure 2.9 the process of generating the feature maps and pooling are executed twice and then, the value of the last pooling layer is used as input on a fully connected network (just as Figure 2.8) to perform the process of classification. Since the output of the last pooling are 2D data, the output is vectorized to be used as input of the fully connected layer. To achieve that, all the pooled feature maps are converted into a vector and concatenated.

As said before, the process of feed forward and backpropagation are also used in CNNs and are performed in similar way as the Fully Connected Networks. For the forward pass step, the image is going through the network and executed over each step to the last layer where an output will be given and a function error E is calculated using the output and the target value. The backpropagation step will derive E in function of each weight and bias of the CNN (which now are the entries of the kernel used in the convolutional layers) and these weights and biases are going to be updated by a given *learning rate* α. An important detail on the backpropagation step is that the pooling operation reduces the size of the feature maps. Therefore, the pooled feature maps are upsampled to match the size of the feature maps generated. With that, the backpropagation can be executed through the pooling layers.

Chapter 3

Related Works

This chapter will go through some previous works that studied methods to recognize and classify diseases. More specifically, various methods that use image processing and visual findings to aid in the diagnosis of diseases are explored and, moreover, methods that work exclusively with Computed Tomography of Lungs are cited.

3.1 Studies on Medical Imaging Analysis

Medical imaging analysis is a broad field of study in the context of Computer Vision and it has been explored in various ways. Previous works have used different kinds of medical images as a way to aid the diagnosis of different diseases and conditions. The appropriate kind of medical image will vary based on the problem being solved. In [M. et al., 2017], X-ray images were used to characterize bone structure using HOG as a way to distinguish osteoporotic cases. In [Lu et al., 2018], Gabor filters were used to predict bone fracture risk using Dual-Energy X-ray Absorptiometry (DXA, or DEXA), which is a medical image type that determines Bone Mineral Density (BMD) by using two x-ray beams.

Other methods may use Optical Coherence Tomography (OCT), a non-invasive medical imaging technique that uses light beams to generate 2-D and 3-D images based on the reflection of these lights. In [Lingley-Papadopoulos et al., 2008], a texture analysis was performed on OCT scans as a way to detect cancer in the urinary bladder. In [Raupov et al., 2016], OCTs were studied as a way to recognize skin cancer using texture analysis methods.

In [Zhang et al., 2008], Magnetic Resonance Imaging (MRI) was used as source of information to discriminate between brain tissues and detect Multiple Sclerosis (MS) lesions using texture classification methods. Early studies have shown that MRI was sensitive in detecting abnormalities in patients with MS [Young et al., 1981].

In the context of diagnosing cancer in patients, the biopsy test is the most common method used. In this method, a sample of the area is extracted from the patient and it is analysed by the pathologist under a microscope. To aid in the process, studies like

[Raupov et al., 2016] have studied Gabor filters as a way to recognize cervical cancer using biopsy images. Computed Tomography (CT) scans have also been studied as a way to diagnose cancer. In [Fan et al., 2018], CT scans were used as a base to sub-classify bladder cancer using statistical approaches.

3.2 Studies on Lungs CT Scans

Computed Tomography is a very common method used on the diagnosis of different conditions in different parts of the body. It consists of an X-ray source and a set of detectors rotating around the patient. With this rotation, the X-ray beam passes through the patient‘s body and is absorbed by the detectors, generating information of different tissues at different angles. This information is processed and the images (scans) are generated. Once it is a very common method, a lot of studies in Computer-aided Diagnosis (CAD) have used this Medical Imaging as a source of information.

More specifically, CTs are very common on diagnosing patients with lung diseases or conditions, because the process allows the lung tissues information to be better highlighted as it generates information for a large amount of slices of the lungs [Pereyra et al., 2014]. In [Mattonen et al., 2014], CT scans were used with some statistical methods using GLCM as a way to predict the recurrence of lung tumor by analysing the texture changes after the patient goes through Stereotactic Ablative Radiotherapy (SABR).

Works like [Sørensen et al., 2008] have explored the usage of the Local Binary Pattern (LBP) as a set features to classify texture in lungs CT scans. In [Song et al., 2012], they combined LBP and HOG as features to recognize a set of different lesions in the lungs using CTs.

Under the focus of a specific set of lung diseases, [Pereyra et al., 2014] has worked with Diffuse Parenchymal Lung Diseases (DPLDs), which is a set of over 150 disorders that are, mainly, characterized by the gradual scaring of lung tissues and frequently leads to respiratory failure. It studied the capability of different characterization methods to recognize these diseases in lungs CT scans.

Chapter 4

Methodology

This chapter presents how the methods described in Chapter 2 were used to characterize the images. Some methods were used to generate a feature vector and some methods are able to characterize the images using its own properties without the need to further extract the features.

4.1 Statistical Methods

4.1.1 Haralick

As stated in Sub-section 2.1.1, the Haralick metrics are a set of features that describe images using the Gray-Level Co-Occurrence Matrix. To characterize the image, the 14 features were used in the classification process [Pereyra et al., 2014].

Therefore, the feature vector extracted from the image is formed by the 14 Haralick descriptors. The process was run over all the images on the dataset and the process of classification was made using the k-Nearest Neighbors using the amount of nearest neighbors as 5. The validation process was made using the Stratified K-Fold Validation with 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

4.1.2 Local Binary Pattern

For this method, the ${LBP_{P,R}}^{riu2}$ operator was used, so the features would have no variation with changes on rotation and luminance of the image using concept of Uniform Pattern (see Sub-Section 2.1.2). To generate the feature vector, let $\mathbf{P} = (P_1, P_2, \ldots, P_l)$ be a vector of l values of P and $\mathbf{R} = (R_1, R_2, \ldots, R_l)$ be a vector of l values of R. For each value of P_k and R_k, a result image LBP_{P_k,R_k} is generated and 2 features [Giakoumoglou, 2021] are extracted: the energy and entropy. Let $LBP_{P_k,R_k}(x, y)$ be the pixel (x, y) of the result image, the two features are given by

$$\textbf{Energy} = \sum_i \sum_j \{ LBP_{P_k,R_k}(i,j) \}^2 \tag{4.1}$$

$$\textbf{Entropy} = -\sum_i \sum_j LBP_{P_k,R_k}(i,j) \log \left(LBP_{P_k,R_k}(i,j) \right). \tag{4.2}$$

In the experiments, the vectors $\mathbf{P} = (8, 16, 24)$ and $\mathbf{R} = (1, 2, 3)$ were used [Ojala et al., 2002]. Since $l = 3$, this will result in a feature vector of length $l \cdot 2 = 6$. The feature vectors were generated for all the images and a k-Nearest Neighbors classification algorithm was executed using $k = 5$. For the validation, the Stratified 10-Fold Validation was used. The results presented on Chapter 5 are resultant of 100 executions.

4.1.3 Histogram of Oriented Gradients

The method described in Sub-Section 2.1.3 was used with the cells size being 8×8 pixels and the block size being 3×3 cells [Dalal and Triggs, 2005, Song et al., 2012]. To normalize the histogram of the cells for each block, the method used the $L2$-norm. It used 9 orientations to generate the histogram, so, the histograms for each cell has 9 bins [Giakoumoglou, 2021].

Using these parameters, the feature vector generated for images of size 64×64 was of length 2916. This vector was generated for each image in the dataset and the classification process was performed using the k-Nearest Neighbor methods with parameter $k = 5$. The validation was performed using the Stratified k-Fold Validation with a total of 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

4.2 Spectral Methods

4.2.1 Law's Texture Energy Measure

For this method, as described in Sub-Section 2.2.1, a family of filters are passed across the image and then measures can be extracted from the result images. Empirically, the family of filters used have length $l = 5$ [Vince et al., 2000, Wu et al., 1992], and, therefore, the family of filters is listed in Equation 2.13. With these filters, the feature vector [Giakoumoglou, 2021] was generated by measuring the energy of the:

1. Result image from the LL filter;
2. Result image from the EE filter;
3. Result image from the SS filter;
4. Average of the result image from the LE filter and EL filter;

5. Average of the result image from the ES filter and SE filter;

6. Average of the result image from the LS filter and SL filter;

Therefore, for each image, a feature vector of length 6 was generated. The classification step was performed using the k-Nearest Neighbors with $k = 5$ and the validation was performed using the Stratified k-Fold Validation methods with 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

4.2.2 Gabor

As stated in Sub-Section 2.2.2, the response to the Gabor filter is $G_{\lambda,\theta}(x, y)$ for a given wavelength λ, frequency $1/\lambda$ and orientation θ. For the experiments, the values of frequency were $1/\lambda = (1/16, 1/18, 1/26, 1/32)$ and the values of orientation were $\theta = (0, \pi/6, 2\pi/6, 3\pi/6, 4\pi/6, 5\pi/6)$ [Lu et al., 2018]. Therefore, the filters used can be seen in Figure 2.2. All the possible combinations of these parameters gives a total of 24 response images $G_{\lambda,\theta}(x, y)$.

With these response images, the Gabor energy $E_{\lambda,\theta}(G)$ of the 24 images are taken using the Equation 2.19. The feature vector is formed by these 24 Gabor energy values. The feature vectors of all images are taken and a classification process is executed using the k-Nearest Neighbors methods with $k = 5$. The validation is executed using the Stratified k-Fold validation using 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

4.2.3 Fourier Transform

For each image, the Fourier Transform was used to generate the frequency information using the process stated in Sub-Section 2.2.3. To apply this information on a texture characterization process, three sub-images were generated from the Fourier transform: information in a vertical band around the origin, information in a horizontal band around the origin and information in a circumference around the origin [Lingley-Papadopoulos et al., 2008]. This step can be seen on Figure 4.1.

Given the three sub-images, the energy was calculated for each one. The feature vector of the input image is formed by these 3 features: the energy of different bands of frequency of the Fourier Transform image. The classification was executed using the k-Nearest Neighbors with $k = 5$ and the validation was executed using the Stratified k-Fold Validation with 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

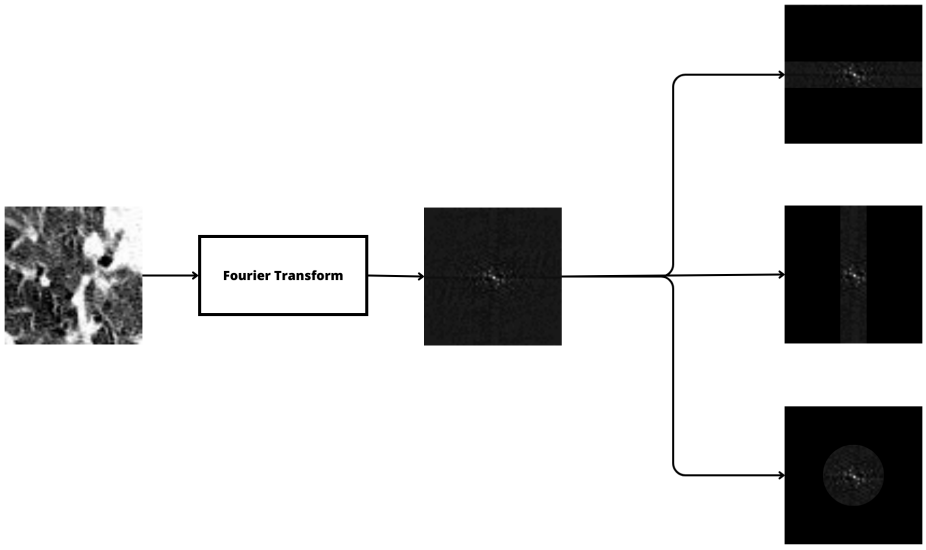


Figure 4.1: The process of generating the 3 sub-images from the Fourier Transform. From top to bottom: horizontal band, vertical band and circumference band.

4.2.4 Wavelet Transform

Using the definitions of the Discrete Wavelet Transform (DWT) and Stationary Wavelet Transform (SWT) stated in the Sub-Section 2.2.4, a method to describe texture can be implemented. The amount of levels used for both DWT and SWT was 3 [Tsiaparas et al., 2011], so the image will be downsampled 3 times, for the DWT, and the filters will be upsampled 3 times, for the SWT.

The wavelet used was the Wavelet Biorthogonal 3.3 (*bior3.3*), from the family of Biorthogonal wavelets and presents symmetricity and is not orthonormal. Being biorthogonal means the wavelet transform associated can be inverted. This wavelet curve can be seen on Figure 4.2.

To generate the feature vector, the mean and energy [Giakoumoglou, 2021] were extracted from each *detail* coefficients for each level for both DWT and SWT. Since there are 3 levels of decomposition and 3 detail coefficient sub-images are generated for each level, the feature vector formed by the mean and energy of these sub-images will have length $= 3 \cdot 3 \cdot 2 = 18$. The feature vector was generated for each image on the dataset and the classification was performed by a k-Nearest Neighbors algorithm with $k = 5$. For the validation, the Stratified k-Fold Validation was executed with 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

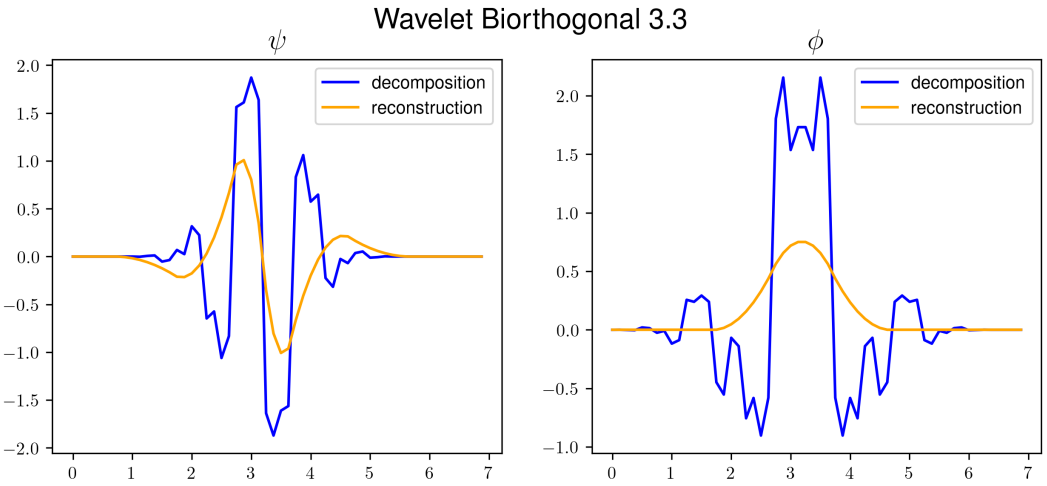


Figure 4.2: The Wavelet Biorthogonal 3.3 components. On the left, the wavelet function ψ. On the right, the scaling function ϕ. Both decomposition and reconstruction versions of the function are plotted.

4.2.5 Fractal Dimension Texture Analysis

As described in Sub-Section 2.2.5, a feature vector MF (see Equation 2.27) can be generated using the H parameter at different resolutions of the image I. The resolutions chosen was the maximum possible [Wu et al., 1992], so, since the images have size 64×64, the MF generated by the method is going to be $MF = (H^{(6)}, H^{(5)}, H^{(4)}, H^{(3)}, H^{(2)})$ with length=5. The feature vector was extracted from all the images and the classification was executed using k-Nearest Neighbors with $k = 5$. The validation was performed using the Stratified k-Fold Validation using 10 folds. The results presented on Chapter 5 are resultant of 100 executions.

4.3 Graph Methods

The methods described in this section use concepts of Graph Theory to implement texture classification process in images.

4.3.1 Shortest Paths

This method uses concept of shortest paths between nodes of a graph to generate the feature vectors of the image [de Mesquita Sa et al., 2013]. It considers the shortest paths along 4 directions of the image: 0°, 45°, 90° and 135°. This means it calculates the shortest paths among all the vertices and chooses the shortest ones along each orientation to generate a representation. These 4 sets of shortest paths can be visualized in Figure 4.3.

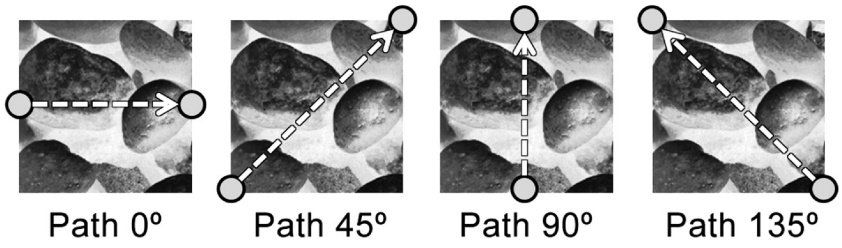


Figure 4.3: Example of the four sets of starting and ending vertices considered in the calculus of the shortest paths. Source: [de Mesquita Sa et al., 2013]

Initially, the image I of size $M \times N$ is converted to a graph $G = (V, E)$ structure where each pixel (x_i, y_i) is a vertex $v_i \in V$. The set of edges E is defined by the *Chebyshev* spatial distance of the pixels $E = \{(v_i, v_j) \in V \times V \mid \max(|x_i - x_j|, |y_i - y_j|) = 1\}$. The weight $w(e)$ of a given edge e is given by $w(e) = |I(x_i, y_i) - I(x_j, y_j)| + \frac{I(x_i,y_i)+I(x_j,y_j)}{2}$, where $I(x, y)$ is the gray-level intensity of the pixel (x, y).

Now, the image is divided into non-overlapping boxes of size $r \times r$, where r is a divisor of the image. For each box, the four shortest paths $(p_{0^\circ}, p_{45^\circ}, p_{90^\circ}, p_{135^\circ})$ are calculated using the Dijkstra's algorithm (see Section 2.4). Now, to represent the image, two vectors are proposed: $\vec{\alpha}_r = [\mu_{0^\circ}, \mu_{45^\circ}, \mu_{90^\circ}, \mu_{135^\circ}]$ and $\vec{\beta}_r = [\sigma_{0^\circ}, \sigma_{45^\circ}, \sigma_{90^\circ}, \sigma_{135^\circ}]$, where μ_{d° and σ_{d° represent, respectively, the mean and standard deviation of the shortest path along the direction d° through all the $r \times r$ boxes. These two vectors are concatenated to form the feature vector $\vec{\psi}_r = [\vec{\alpha}_r, \vec{\beta}_r]$. This process is executed over different $r \times r$ sizes of boxes, as it is possible to see in Figure 4.4.

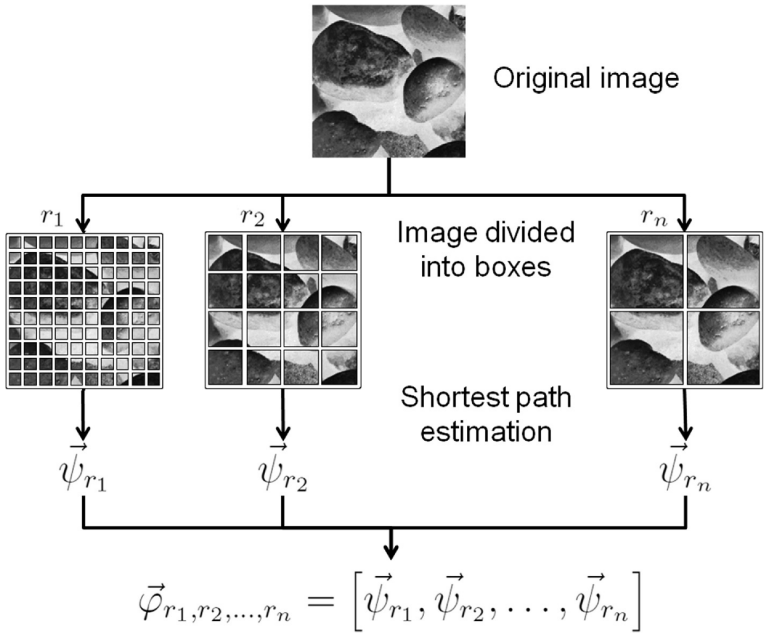


Figure 4.4: Overview of the process of feature vector extraction, where r_n is the size of squares in which the image is divided into. Source: [de Mesquita Sa et al., 2013].

Let the n boxes sizes chosen be $\mathbf{r} = (r_1, \ldots, r_n)$, the final feature vector that characterizes the original image is defined as the concatenation of all $\vec{\psi}_{r_i}$. Therefore the final feature vector is $\vec{\varphi}_{\mathbf{r}} = [\vec{\psi}_{r_1}, \ldots, \vec{\psi}_{r_n}]$. In the experiments, since the boxes sizes r need to be divisor of the 64×64 image, the chosen r were $\mathbf{r} = (4, 8, 16, 32)$. This gives a total of 32 features. This process was executed over each image and the classification algorithm used was the k-Nearest Neighbor with $k = 5$. For the validation, it was used the Stratified k-Fold Validation with 10 folds.

4.3.2 Diffusion in Networks

Proposed in [Gonçalves et al., 2016], this method uses concepts of diffusion and multiple graphs to characterize images. The main steps can be seen on Figure 4.5.

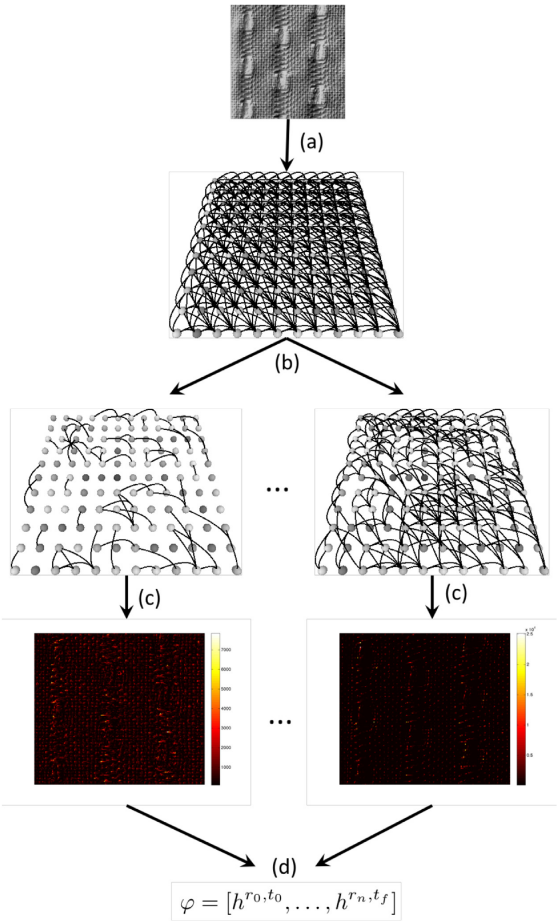


Figure 4.5: Main steps of the proposed method: (a) image is mapped into a regular directed network; (b) links are removed based on the pixel intensity difference; (c) activity of each node is estimated by random walks; (d) histograms of activity are calculated to describe the image. Source: [Gonçalves et al., 2016].

Initially, the image I of size $M \times N$ is converted to a graph $G = (V, E)$ structure where each pixel $p_i = (x_i, y_i)$ is a vertex $v_i \in V$. The set of edges E is defined by the

Euclidean distance between the pixels of the image. If the distance is within a certain radius r, its weight is set as the difference of intensity, otherwise, the nodes corresponding to the pixels are not connected:

$$e_{v_i,v_j} = \begin{cases} I(p_i) - I(p_j), & \text{if } dist\,(p_i, p_j) \leq r \\ NaN, & \text{otherwise} \end{cases}, \tag{4.3}$$

where $dist(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ and $I(p_i)$ is the intensity of the pixel in the original image (Figure 4.5 (a)). To transform this network G in a directed network, all the edges with negative weight $e_{v_i,v_j} < 0$ are discarded. Given the directed graph, a function $\phi(t, N)$ is proposed to perform a limitation on the connections of the network and, therefore, to present different texture information for different threshold t using

$$e_{v_i,v_j} = \begin{cases} e_{v_i,v_j} & \text{if } 0 < e_{v_i,\nu_j} \leq t \\ NaN, & \text{otherwise} \end{cases}. \tag{4.4}$$

As t decreases, the network captures more fine details on the image because only pixels with slight intensity difference are connected. For larger values of t, more differences on intensities are considered and more global information is represented by the network (Figure 4.5 (b)). Given the network generated by this process, the activity of each node is extracted. For this, it is used the random walker probability, where the probability of a walker in node v_i go to another node v_j is proportional to the weight of the edge that connects these nodes:

$$p(v_i, v_j) = \frac{e_{v_i,v_j}}{\sum_{v_k \in V} e_{v_i,v_k}}. \tag{4.5}$$

The random walker takes W walks starting on each node and the activity $\alpha(v_i)$ is the amount of times the walker passed by the v_i node. This generates an activity image for each combination of radius r and threshold t (Figure 4.5 (c)). To generate the feature vector for each activity image, an association with the in-degree ${k_{v_i}}^{in}$ (See Section 2.4) of each node is made to calculate the histogram $h^{r,t}(j)$ using

$$h^{r,t}(j) = \sum_{v_i \in V} \begin{cases} \alpha^{r,t}\,(v_i), & \text{if } k_{v_i}^{in} = j \\ 0, & \text{otherwise} \end{cases}, \tag{4.6}$$

where $\alpha^{r,t}\,(v_i)$ is the activity of a node v_i on the graph generated by radius r and threshold t. The higher the in-degree of a node, higher the chance of it being visited. But it is not always true because the activity uses a random walker based on the weight of the edges. Therefore, this histogram can represent the textures properties of the image. The histograms for each combination of radius r and threshold t are generated and concatenated to form the feature vector φ that represents the image (Figure 4.5 (d)).

As said before, this is executed for a set of values of radius $\mathbf{r}$ and a set of values of

threshold $\mathbf{t}$. For the experiments, $\mathbf{r} = (\sqrt{2})$ and $\mathbf{t} = (130, 190, 250)$ were used. This gives a feature vector of length $l = 1 \cdot 3 \cdot {k^{in}}_{max}$, where ${k^{in}}_{max}$ is the maximum in-degree on the images. Empirically, the value of ${k^{in}}_{max}$ was found to be 9. So, the feature vector for each image φ has length 27. These feature vectors were used in a classification step using the k-Nearest Neighbors algorithm using $k = 5$ and the validation was performed using the Stratified k-Fold Validation with 10 folds.

4.4 Deep Learning Methods

4.4.1 Texture Convolutional Neural Network (T-CNN)

Proposed in [Andrearczyk and Whelan, 2016], this method uses a specific architecture of Convolutional Neural Network (CNN) responsible for extracting textural information of the input image. Its simple structure promotes good performance while maintaining a low number of trainable parameters. The method is called Texture Convolutional Neural Network (T-CNN). Similar to regular CNN, it is composed of Convolutional Layers, Pooling and Fully connected layers (see Section 2.4). However, the key difference is in the step of vectorization of the feature maps to the connected layers. The pooling is chosen in a way that each feature map is transformed into a single value. Therefore, after this step, the output will already be a one dimensional vector.

The T-CNN were proposed as a family of Neural Networks where the difference between each variation is the amount of convolutional layers in the architecture. In the experiment, the method was tested with five configurations T-CNN-1 to T-CNN-5. Each one had 1 to 5 convolutional layers, respectively. In Figure 4.6, it is presented the T-CNN-2 architecture.

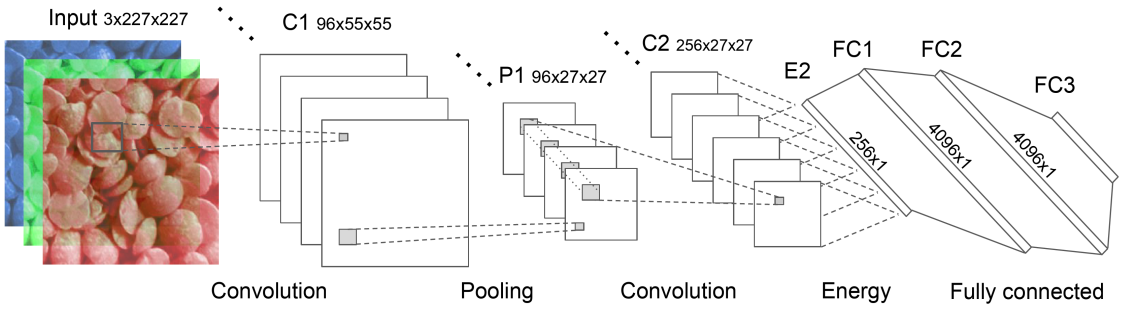


Figure 4.6: T-CNN architecture using two convolution layers. Source: [Andrearczyk and Whelan, 2016].

The architecture initiates with a convolution layer C1 with 96 kernels. It performs the normalization and it passes the result through the $ReLU$ activation function. Then it goes through a max pooling operation at P1. After that, there is the second convolution

layer C2 and an activation using the ReLU function. The next layer is the E2, where the energy of the feature maps from C2 is extracted using an Average Pooling with pool size equal to the feature map size to generate only one value as output. In Figure 4.6, the size of the pool is 27×27. This concept comes from the use of energy measure from filters response in methods that has fixed filters. In the context of CNNs, these filters are learned through the learning process of the network.

Since the output of E2 is a one dimensional vector, it can be easily connected to the dense layers. E2 is followed by 3 fully connected layers: FC1 and FC2 as hidden layers and FC3 as the output layer with the classification of the texture.

In [Andrearczyk and Whelan, 2016] the architecture with best overall performance was T-CNN-3, so it was the one chosen to be used in the experiments. Its layers are the same as T-CNN-2, with the addition of one extra convolution layer before the Energy extraction layer. The network was trained from scratch over the dataset using the split of the data in training, validation and testing set. It was used the Adam optimizer with learning rate of 10^{-4}.

4.4.2 Wavelet Convolutional Neural Networks

The method was proposed in [Fujieda et al., 2018] using concepts of Wavelet Transform applied to a Convolutional Neural Network to perform a multiresolution analysis of images. The regular CNNs can be viewed as a limited form of multiresolution analysis, but only by applying the concepts of decomposition of wavelet transformation (see Sub-Section 2.2.4) it presents a characteristic hierarchical decomposition.

To achieve this, [Fujieda et al., 2018] proposes a pair of filters $k_{h,t}$ and $k_{l,t}$ that are *high-pass* and *low-pass* filters, respectively, at a given level t. On the context of Wavelet Transform, $k_{h,t}$ is the wavelet function and $k_{l,t}$ is the scaling function. When compared to regular CNNs that use filters k_t learned iteratively through the dataset, the Wavelet CNN presents better computational performance once the filters are differentiated between high-pass and low-pass. The Wavelet function used in the experiments was the *Haar wavelets*, but the model is not restricted. The architecture of the Wavelet CNN with 4 levels of decomposition can be seen in Figure 4.7.

On the first level, the input image goes through the two filters $k_{h,1}$ and $k_{l,1}$. The response of each filter is concatenated and goes through a regular convolution layer with filter k_1: firstly through 64 filters and then another 64 filter convolution with stride of 2 (so it reduces the size of the feature maps by 2). At the same time, the response of $k_{l,1}$ goes trough the second level decomposition $k_{h,2}$ and $k_{l,2}$. The output of these filters a concatenated and passed by a 64 filters convolution. This output and the output of the convolution k_1 are concatenated and forms the input of the second convolution layer k_2, that uses two convolution process with 128 filters and the second one with stride of 2 to

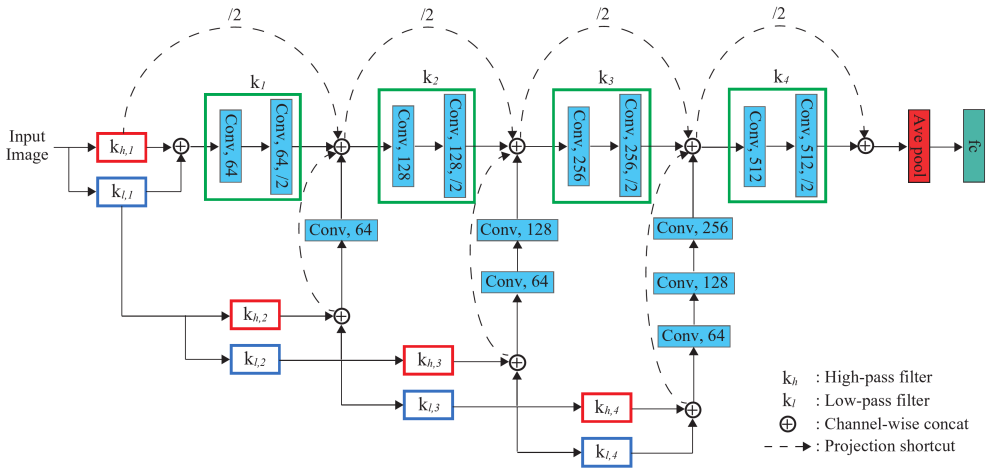


Figure 4.7: Overview of wavelet CNN with 4-level decomposition of the input image. Source: [Fujieda et al., 2018].

reduce the size of the feature maps. This process goes on until 4 levels of decomposition is reached. The k_4 output is then vectorized by an *average pooling* operation. The final layers f_c are composed by two fully connected layers with operations of normalization and *ReLU* activation and an output layer that gives the final classification of the input texture image. The network was trained from scratch over the dataset using the split of the data in training, validation and testing set. It was used the Adam optimizer with learning rate of 10^{-3}.

4.5 Hybrids

In this section, methods that use knowledge from multiple areas are described. They are called hybrid because their behavior and performance are dependent of multiple concepts reported in Chapter 2.

4.5.1 PCA Network

The PCA Network (PCANet) was proposed in [Chan et al., 2015] and it uses concepts of Principal Component Analysis (Section 2.3) over the training images to generate filters that best describe images in a deep network architecture. An illustration showing the main components of the network can be seen in Figure 4.8.

The image is passed on the first stage of PCA filters which generates L_1 output images. Then, each of these images are passed through the second stage of PCA filters, generating $L_1 \cdot L_2$ output images. The next step is to binarize and convert the binary information to decimal data. The final step generates an histogram that characterizes the input image. A more detailed look on the architecture can be seen on Figure 4.9.

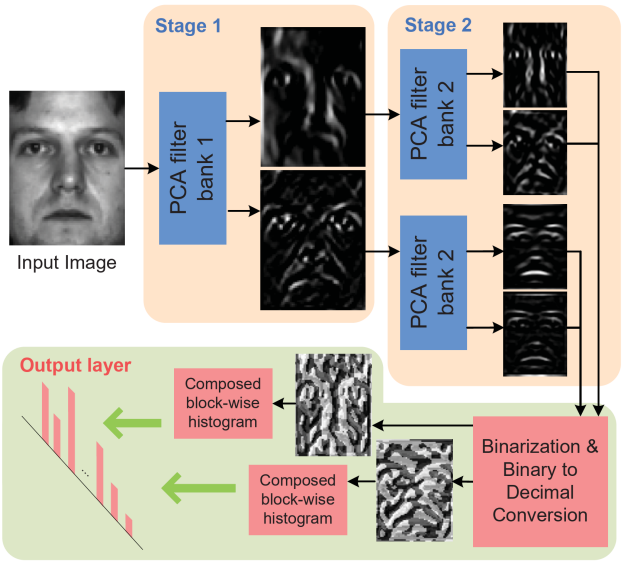


Figure 4.8: Illustration of how the proposed PCANet extracts features from an image through the three simplest processing components: PCA filters, binary hashing, and histograms. Source: [Chan et al., 2015].

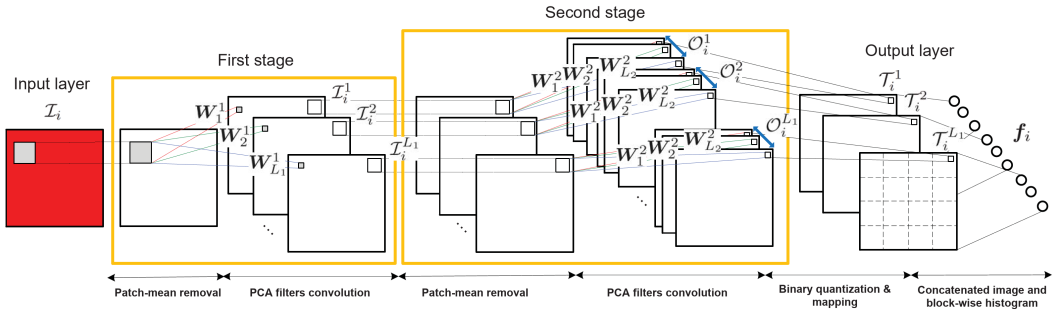


Figure 4.9: A detailed block diagram of the proposed (two-stage) PCANet. Source: [Chan et al., 2015]

Let N be the amount of training images $\{\mathcal{I}_i\}_{i=1}^{N}$ of size $m \times n$ and at each stage the 2D filters are of size $k_1 \times k_2$. Those filters are learned from the training images using PCA concepts. For the first stage, a patch of size $k_1 \times k_2$ is taken around each pixel of the image and it is vectorized as $\mathbf{x}_{i,j} \in \mathbb{R}^{k_1 \cdot k_2}$ where j is the j-th vectorized patch of the image $\mathcal{I}_i$. Then, each of these vectors are subtracted by the patch mean and generate $\bar{X}_i = [\bar{\mathbf{x}}_{i,1}, \bar{\mathbf{x}}_{i,2}, \ldots \bar{\mathbf{x}}_{i,\tilde{m}\tilde{n}}]$, where $\tilde{m} = m - [k_1/2]$ and $\tilde{n} = n - [k_2/2]$. This is executed over all N training images, which gives $\mathbf{X} = [\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_2, \ldots, \bar{\mathbf{X}}_N] \in \mathbb{R}^{k_1 k_2 \times N\bar{m}\bar{n}}$. Let L_i be the amount of filters at stage i, the PCA will minimize the reconstruction error within a family of orthonormal filters [Chan et al., 2015] with

$$\min_{\mathbf{V} \in \mathbb{R}^{k_1 k_2 \times L_1}} \| \mathbf{X} - \mathbf{V}\mathbf{V}^T\mathbf{X} \|_F^2, \quad \text{s.t. } \mathbf{V}^T\mathbf{V} = \mathbf{I}_{L_1}, \tag{4.7}$$

where $\mathbf{I}_{L_1}$ is the identity matrix of size $L_1 \times L_1$. Using the knowledge of PCA, the solution

will be the L_1 principal eigenvectors of $\mathbf{X}\mathbf{X}^T$. Therefore, the L_1 filters of the first stage can be expressed as

$$\mathbf{W}_l^1 \doteq \text{mat}_{k_1,k_2}(\mathbf{q_l}(\mathbf{X}\mathbf{X}^T)) \in \mathbb{R}^{k_1 \times k_2}, \quad l = 1, 2, \ldots, L_1, \tag{4.8}$$

where $\mathbf{q_l}(\mathbf{X}\mathbf{X}^T)$ is the l-th principal eigenvectors of $\mathbf{X}\mathbf{X}^T$, $\text{mat}_{k_1,k_2}(v)$ is an operation that maps the vector $v \in \mathbb{R}^{k_1 \cdot k_2}$ to a matrix $\mathbf{W} \in \mathbb{R}^{k_1 \times k_2}$ and $\mathbf{W}_l^i$ is the l-th filter in the i-th stage.

The process is repeated for the second stage, except the filters are now calculated over the L_1 outputs of the first stage for each of the N training images. The filters of the second stage can be expressed as

$$\mathbf{W}_\ell^2 \doteq \text{mat}\, k_{1,k_2}(\mathbf{q}_\ell(\mathbf{Y}\mathbf{Y}^T)) \in \mathbb{R}^{k_1 \times k_2}, \quad \ell = 1, 2, \ldots, L2, \tag{4.9}$$

where $\mathbf{Y}$ is analogous to X and, therefore, is a matrix $\mathbf{Y} = [\mathbf{Y}^1, \mathbf{Y}^2, \ldots, \mathbf{Y}^{L_1}] \in \mathbb{R}^{k_1 k_2 \times L_1 N \tilde{m} \tilde{n}}$ formed after the vectorization of patches around each pixel on the L_1 images of all N training images.

For each of the L_1 images of the stage 1, L_2 images are the output at stage 2. The binarization is performed using the *Heaviside* function: If the value of the pixel is positive, it is turned into 1; it is turned into 0 otherwise. Now, with this binarization, the L_2 images can be seen as only one image $\mathcal{T}_i^l$ with each pixel being the decimal conversion of L_2 bits. After this step, for each input image $\mathcal{I}_i$, there is L_1 images $\mathcal{T}_i^l$, for $l = 1, \ldots, L_1$.

Now, each of the L_1 images $\mathcal{T}_i^l$ are partitioned into B blocks, and a histogram of 2^{L_2} bins is calculated. The histograms of all the blocks of an image $\mathcal{T}_i^l$ are concatenated into a vector $\text{Bhist}(\mathcal{T}_i^l)$. With the concatenation of all L_1 vectors $\text{Bhist}(\mathcal{T}_i^l)$, a feature vector f_i for the image $\mathcal{I}_i$ can be defined as

$$f_i \doteq [\text{Bhist}(\mathcal{T}_i^1), \cdots, \text{Bhist}(\mathcal{T}_i^{L1})]^T \in \mathbb{R}^{(2^{L2})L_1 B}. \tag{4.10}$$

For the experiments, the size of the filters of each stage was set as $k_1 = k_2 = 5$. The amount of filters on each stage was chosen to be $L_1 = 8$ and $L_2 = 8$. The block B size used to generate the histograms was set as 50×50. These parameters were used to generate the feature vectors f_i of length $L_1 \cdot 2^{L_2} = 2048$ for each image in the training set. For the classification step, the Support Vector Machine (SVM) was used with the *Radial Basis Function* kernel. To validate the performance, the Stratified k-Fold Validation method was used with 10 folds. The results presented on Chapter 5 are resultant of 15 executions.

4.5.2 Proposed Method

The method proposed in this work and in [Álvaro Albuquerque et al., 2022] expands the usage of Complex Networks and Graph Theory concepts presented in

[Gonçalves et al., 2016] while combining with some classical statistical methods (Section 2.1). The overall steps of the method can be seen in Figure 4.10.

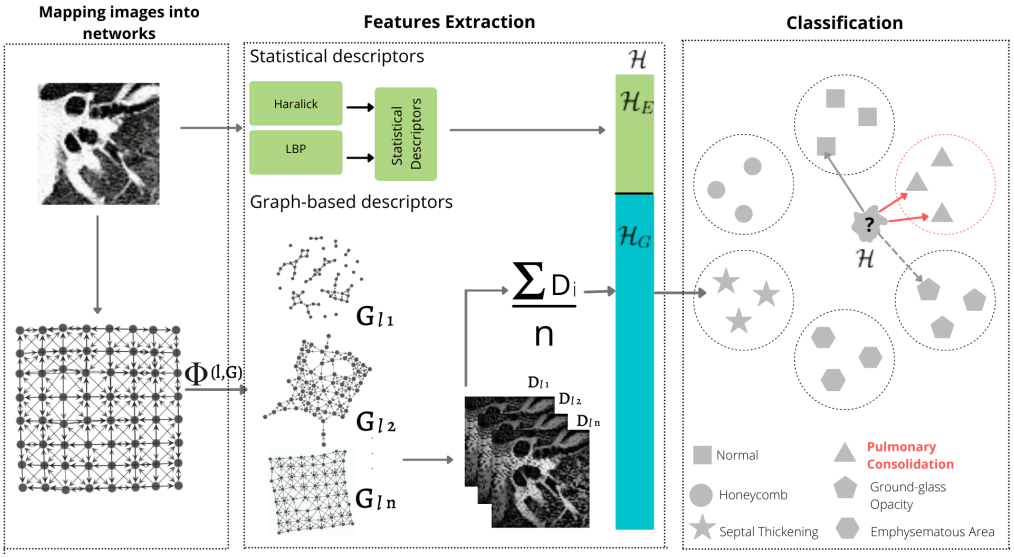


Figure 4.10: Classification of textural images using the Closeness Centrality metric and statistical image texture descriptors. Source: [Álvaro Albuquerque et al., 2022].

The process is divided into 3 main steps: *mapping images into networks*, *features extraction* and *classification*. To map the image into a network, the first step is to model a grayscale image I into a network $G = (V, E)$ based on [Gonçalves et al., 2016] and [Couto et al., 2017], where V is the set of all the vertices and E is the set of all edges. Each pixel $p_i = (x_i, y_i)$ (where x and y represent its spatial position) has an intensity value associated $I(p_i) \in [0, 255]$. The pixel p_i is mapped into a vertex (or node) $v_i \in V$.

Two vertices v_i and v_j are connected if the Euclidean distance $d(p_i, p_j) = \sqrt{(x_i - x_j)^2 - (y_i - y_j)^2}$ is equal or less than a given radius r. For each graph edge $e \in E$, a weight e_{v_i,v_j}, defined by the value of the pixels intensity difference, is assigned, according Equation 4.11. Initially, this mapping is a regular weighted graph presenting connected vertices in a neighborhood defined by the radius r.

$$e_{v_i,v_j} = \begin{cases} I(p_i) - I(p_j), & \text{if } d(p_i, p_j) \leq r \\ NaN \text{ (Not a Number)}, & \text{otherwise} \end{cases} \tag{4.11}$$

In Equation 4.11, the non-existence of an edge is given by the Not a Number (NaN) symbol. Thus, to transform the obtained network into a complex network G, we applied a transformation $\phi(l, G), \phi : G \to \mathcal{G}$ on the edges of the network to reveal the properties of the original image texture, where $\mathcal{G}$ is a multiscale graph set. It consists of selecting an edge according to the value of its weight e_{v_i,v_j}. Edges with weight less than a threshold l, are selected.

To obtain a directed graph, links with negative weights are discarded. In Equation 4.12 it is possible to see that edges with negative weights are excluded from the set of edges E. Therefore, the direction of an edge $e \in E$ in directed graph G is given by the pixel with larger intensity values to pixels with lower intensity values.

$$e_{vi,vj} = \begin{cases} e_{vi,vj}, & \text{if} \quad 0 < e_{vi,vj} \leq l \\ NaN, & \text{otherwise} \end{cases} \tag{4.12}$$

The transformation $\phi(l, G)$ can be seen as a multiscale graph analysis. For each value of l, the original graph is transformed into a l-scaled graph $G_l \in \mathcal{G}$. In this way, small values of l, provides detailed local information about image textures, while larger values of l presents better global information, such as image edges. l is called scaled threshold.

To generate the features, the closeness of each l-scaled graph G_l is taken using Equation 2.33. Let D_l be the matrix where each entry $D_l(v_i)$ is the closeness centrality metric of the vertex $v_i \in V$ in a given graph G_l. Then, the average matrix D is taken as the average of all D_l for all $l \in L$, where $L = \{l_1, l_2, \ldots, l_N\}$ (Equation 4.13).

$$D = \frac{1}{N} \sum_{l \in L} D_l \tag{4.13}$$

The creation of a feature vector using the closeness centrality metric consists of the relation between the intensity values of the image I and the values of the matrix D obtained by applying the centrality closeness measure over the graphs $G_l \in \mathcal{G}$. The value $D(v_i)$ is finally the average of all closeness values for all l-scaled graphs, for the vertex $v_i \in V$, and consequently, for the pixel p_i. Let m be the amount of equally spaced intervals in the intensity image range $[0, 255]$, therefore $[w_k, z_k]$ is the k-th interval into the range. Then, the relation between the intensity values of the image I and the values of the matrix D can be expressed by S_{w_k,z_k} (Equation 4.14)

$$S_{w_k,z_k} = \sum_{v_i \in M_k} D(v_i) \tag{4.14}$$

where $M_k = \{v_i \in V | w_k \leq I(p_i) < z_k\}$.

The process of generating the $\mathcal{H}_G$ can be seen on Figure 4.11. Some pixels with lower intensity levels on the original image are highlighted in red and some pixels of higher intensity level are highlighted in blue. The values of closeness centrality of the correspondent pixels in the Closeness Centrality matrix D are summed based on the intensity levels, resulting in S_{w_1,z_1} for pixels with lower intensity and S_{w_m,z_m} for pixels with higher intensity.

Then, for a single image, the feature vector $\mathcal{H}_G = \{S_{w_1,z_1}, S_{w_2,z_2}, \ldots, S_{w_m,z_m}\}$ can be defined. The $\mathcal{H}_E$ is formed by the 14 Haralick's features (see Sub-Section 2.1.1) and by a 10 bin histogram of the LBP image result (see Sub-Section 2.1.2). The feature vector $\mathcal{H}$

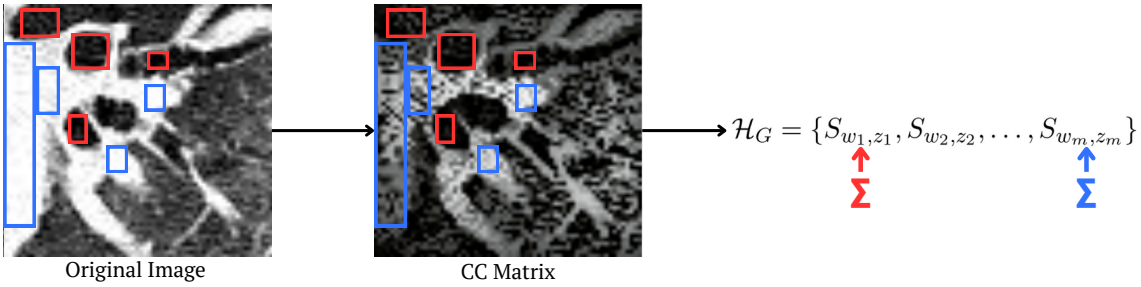


Figure 4.11: $\mathcal{H}_G$ generation.

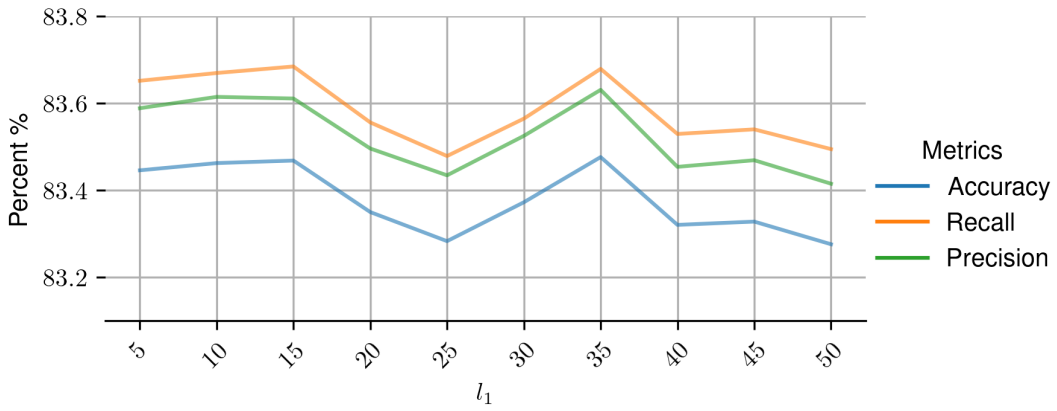


Figure 4.12: Initial threshold l_1 hyperparameter analysis. The hyperparameter l_1 were incremented by 5 until reach 50.

that characterize the image is the concatenation of $\mathcal{H}_E$ and $\mathcal{H}_G$.

The parameters of the method are: the radius r used to connect the vertices; the initial threshold l_1, which is the first value of the set L; the increment amount l_i and the amount of times the increment is used n, that will be used to generate the other values of L. According to [Gonçalves et al., 2016], the main parameters to build the graph are the radius r and the initial threshold l_1. This occurs because the radius is the parameter that defines the connection area of each vertex and the initial threshold limits the approach's analysis scale. Therefore, given the importance of these parameters, we ran experiments to find the best values for them.

On Figure 4.12 we can see the algorithms performance as we increment l_1. The values of $l_1 = 15$ and $l_1 = 35$ stand out once they present better accuracy, recall and precision when compared to other values. With $l_1 = 15$ we reach accuracy of 83.46%, precision of 83.61% and recall of 83.68% while with $l_1 = 35$ the accuracy is 83.47%, precision is 83.63% and recall is 83.67%. Given these metrics, we decided to run the experiments with $l_1 = 15$, once $l_1 = 35$ presents a peak along its neighborhood and can bring more uncertainty to the algorithm.

On Figure 4.13 we can observe that the approach's performance tends to decrease as we increase the radius of the algorithm. Therefore, we used radius $r = 2$ to run the

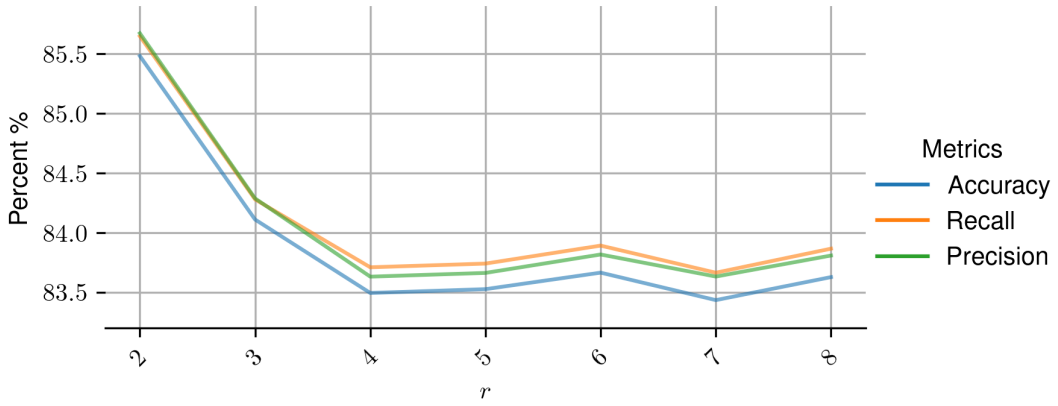


Figure 4.13: Radius r hyperparameter analysis. The hyperparameter r were analyzed in the range of 2 to 8.

experiments once it presents the best metrics: accuracy of 85.48%, precision of 85.67% and recall of 85.64%. We believe that the smaller the radius, the better is for the method to detect variations in the images textures and, therefore, favoring the classification.

Given this analysis, the proposed approach was executed using the following parameters: an initial threshold, $l_1 = 15$; threshold increment $l_i = 40$; amount of times the threshold is incremented $n = 5$; and the radius defining the neighbourhood of vertex v_i, $r = 2$. The threshold increment is used to generate multiple scale networks. Larger values of l_i allows a better exploration of the graph's scales [Gonçalves et al., 2016]. All these parameter values were chosen based on experiments ran with the objective to best describe the computed tomography images of the DPLD.

To generate the vector $\mathcal{H}_G$ for each image, 75 equally partitioned intervals were used with the normalized range $[0, 1]$. Then, the 75 intervals were: [0,1/75), [1/75, 2/75), ..., [74/75, 1]. These intervals will characterize the matrix D. With these parameters, the length of $\mathcal{H}_G$ is 75. The LBP operator was the ${LBP_{P,R}}^{riu2}$ with $P = 24$ and $R = 3$ which, with the extraction of the histogram of 10 bins and the 14 haralick descriptors, makes the $\mathcal{H}_E$ have length 24. Therefore, the feature vector $\mathcal{H}$ has length 99.

This feature vector was extracted from all the images in the dataset and then used in the classification step with the k-Nearest Neighbors algorithm with $k = 5$. The results were validated using the Stratified k-Fold Validation with 10 folds.

Chapter 5

Experimental Results and Discussions

5.1 Dataset

The dataset used in the experiments presented in this document is composed of textural images of computed tomographies of the lungs region in different patients. This data was extracted from exams conducted in the Clinical Hospital of the Ribeirão Preto School of Medicine – University of São Paulo (HCFMRP – USP). The initial database consisted of 246 High-Resolution Computed Tomographies (HRCTs) from 108 different exams selected by a group of radiologists. These images were grouped based on their radiology report and radiographic pattern, which generated categories of approximately 35 images. Only 6 of these categories were selected: *honeycomb*, *ground glass*, *septal thickening*, *pulmonary consolidation*, *emphysematous areas* and *healthy*.

Then, those images were normalized to pixels of 8 bits and, to highlight the patterns, a histogram equalization and centering was performed. With binary thresholding and object selection, the Regions of Interest (ROI) were extracted. This process generated a total of 3258 ROIs of size 64×64 partitioned into the six previously mentioned categories. Table 5.1 shows the support of each class in the database. More details of the dataset generation process can be seen in [Pereyra et al., 2014]. Examples of the ROIs of each class can be seen in Figure 5.1.

5.2 Results

To evaluate the performance of each method presented in Chapter 4, the values of accuracy, precision and recall were extracted. Furthermore, the amount of features that the method generates was also noted once it can impact the algorithm's computational cost.

Given a dataset with elements of two possible classes: $\mathcal{C}$ and, its negative, $\tilde{\mathcal{C}}$. The set

Class	**Support**
Healthy	590
Pulmonary Consolidation	451
Emphysematous Area	502
Septal Thickening	590
Honeycomb	530
Ground-glass Opacity	595
Total	**3258**

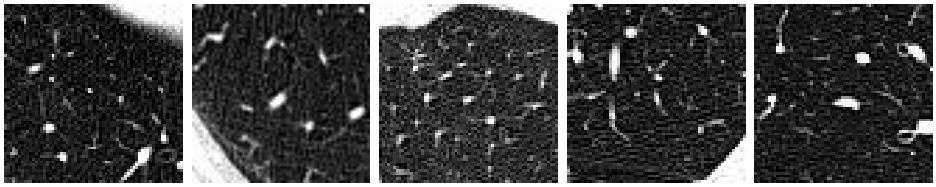
Table 5.1: Support of all the classes in the database.

of *true positives* (TP) is formed by the elements the algorithm correctly classified as $\mathcal{C}$ and the set of *false positives* (FP) is formed by the elements the algorithm classified as $\mathcal{C}$ but are $\tilde{\mathcal{C}}$. The *true negatives* (TN) set is formed by the elements the algorithm correctly classified as $\tilde{\mathcal{C}}$ and the *false negatives* (FN) set is formed by the elements the algorithm classified as $\tilde{\mathcal{C}}$ but are actually from class $\mathcal{C}$.

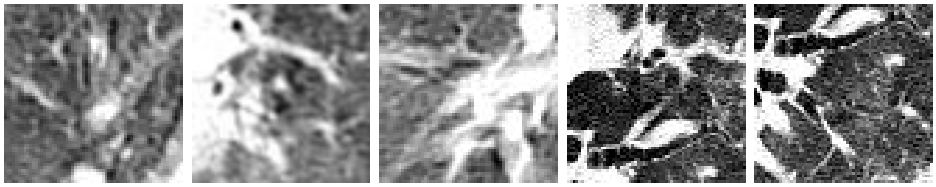
Accuracy scores the method in a more general point-of-view once it refers to how close the algorithm is to the truth and is given by $\frac{TP+TN}{TP+TN+FP+FN}$. Precision and recall are commonly used to analyze classification algorithms once it measures the performance of the method under the optics of the classes. Precision can be viewed as "the amount of right guesses out of all the guesses of the algorithm for a given class". Recall, however, can be simplified as "the amount of individuals the algorithm was able to correctly classify out of all individuals of the given class". Mathematically, precision is given by $\frac{TP}{TP+FP}$ and recall is given by $\frac{TP}{TP+FN}$.

On Table 5.2 it is possible to see the results for the statistical methods of Section 4.1. Haralick's method was capable of achieving the highest accuracy, recall and precision among the methods of this context with, respectively, 79.32%, 79.63% and 79.65%. LBP came in second with accuracy of 54.16%, recall of 54.88% and precision of 55.55% and HOG came in third with the lowest score of 23.70%, 22.43% and 31.07% for accuracy, precision and recall. It is important to notice the considerate difference of the Haralick's method when comparing the amount of feature of each method. While Haralick has the best performance with a low amount of 14 features, the HOG method generates 2916 features with a very low performance. Therefore, for the context of the purely statistical methods analysed in this work, the Haralick method presents better results for the dataset used.

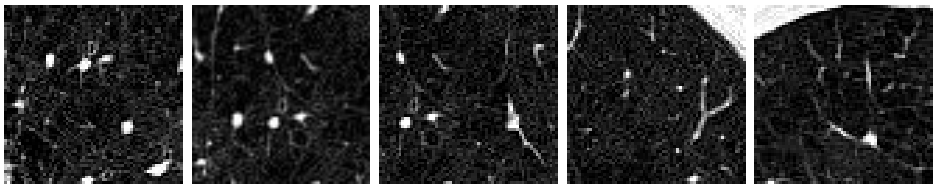
For the methods that use spectral information to classify images described in Section 4.2, the results are presented in Table 5.3. The method that had the highest score was the Stationary Wavelet Transform (SWT) with accuracy of 73.38%, recall of 73.98% and precision of 73.86%. The Law's Texture Energy Measure came in second with accuracy, recall and precision of, respectively, 58.87%, 59.77% and 59.90%. The Discrete Wavelet



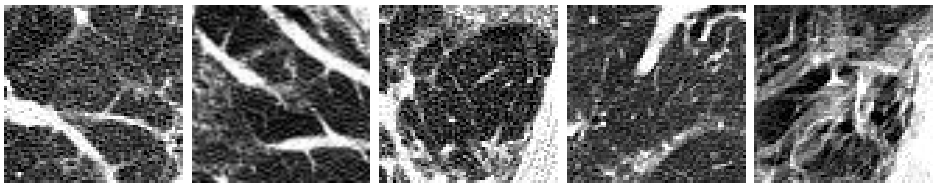
(a) Healthy Lung



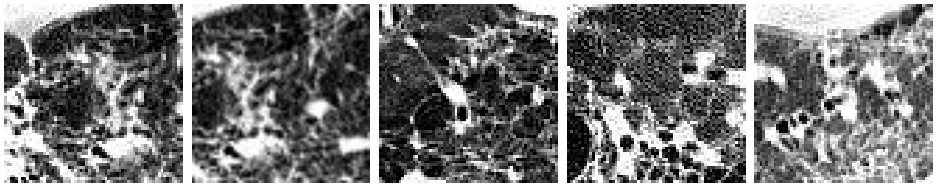
(b) Pulmonary Consolidation



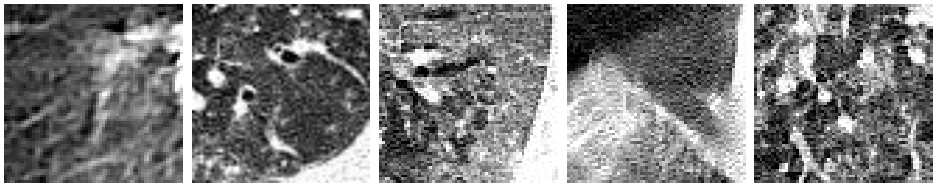
(c) Emphysematous Area



(d) Septal Thickening



(e) Honeycomb



(f) Ground-glass Opacity

Figure 5.1: Regions of Interest presenting texture patterns of DPLDs in high-resolution CT images (Adapted from [Pereyra et al., 2014]).

Transform is in third place with accuracy, recall and precision of, respectively, 52.21%, 52.70% and 52.26%. The Fractal Dimension came in fourth, followed by the Fourier Trans-

Method	**Features**	**Acc.**	**Rec.**	**Prec.**
LBP (sub-section 4.1.2)	6	54.16%	54.88%	55.55%
Haralick (sub-section 4.1.1)	14	79.32%	79.63%	79.65%
HOG (sub-section 4.1.3)	2916	23.70%	22.43%	31.07%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.2: Accuracy, Recall and Precision for Statistical Methods (Section 4.1) analysed.

Method	**Features**	**Acc.**	**Rec.**	**Prec.**
Law's (sub-section 4.2.1)	6	58.87%	59.77%	59.90%
Gabor (sub-section 4.2.2)	24	38.03%	38.33%	38.72%
Fourier (sub-section 4.2.3)	3	47.87%	48.37%	49.47%
DWT (sub-section 4.2.4)	18	52.21%	52.70%	52.26%
SWT (sub-section 4.2.4)	18	73.38%	73.98%	73.86%
Fractal Dimension (sub-section 4.2.5)	5	49.80%	50.36%	50.90%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.3: Accuracy, Recall and Precision for Spectral Methods (Section 4.2) analysed.

form and then the Gabor Filters. The amount of features were, overall, pretty low. The difference in the performance of the methods DWT and SWT may be due to the translation invariance the SWT variation provides to the Wavelet Transform method. Therefore, among the spectral methods analysed, the Stationary Wavelet Transform presented the best performance while keeping the amount of features in a low amount of 18.

In the context of the methods that use graph theory to classify textures described in Section 4.3, the results can be seen on Table 5.4. It is possible to notice that the Diffusion in Networks method had the better results with accuracy, recall and precision of 77.00%, 77.26% and 77.51% when compared to the Shortest Paths method with 70.55%, 71.03% and 71.07%. The amount of features both method generates are very close: Diffusion in Networks generates 27 features and Shortest Paths method generates 32 features. With this, it is possible to notice that the Diffusion in Networks method, which made use of the network's diffusion and the amount of activity of the nodes based on a Random Walker, was capable of better describing the textures in the dataset used.

When comparing the methods that use Deep Learning to classify textures referenced in Section 4.4, the results can be seen on Table 5.5. The T-CNN-3 was capable of achieving

Method	**Features**	**Acc.**	**Rec.**	**Prec.**
Shortest Paths (sub-section 4.3.1)	32	70.55%	71.03%	71.07%
Diffusion in Networks (sub-section 4.3.2)	27	77.00%	77.26%	77.51%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.4: Accuracy, Recall and Precision for Graph Methods (Section 4.3) analysed.

Method	**Acc.**	**Rec.**	**Prec.**
T-CNN-3 (sub-section 4.4.1)	81.37%	80.26%	82.24%
WaveletCNN (sub-section 4.4.2)	68.75%	61.43%	73.91%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.5: Accuracy, Recall and Precision for Deep Learning Methods (Section 4.4) analysed.

Method	**Features**	**Acc.**	**Rec.**	**Prec.**
PCANet (sub-section 4.5.1)	2048	81.35%	80.96%	82.33%
Proposed Method (sub-section 4.5.2)	99	85.45%	85.62%	85.63%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.6: Accuracy, Recall and Precision for Hybrid Methods (Section 4.5) analysed.

81.37%, 80.26% and 82.24% of accuracy, recall and precision, respectively. This was superior to the WaveletCNN that achieved 68.75%, 61.43% and 73.91% of accuracy, recall and precision.

On Table 5.6 the results for the Hybrid methods described in section 4.5 are displayed. The PCANet method got accuracy of 81.35%, recall of 80.96% and precision of 82.33% with a total of 2048 features. A better performance was noted on this work's proposed method. It presented an accuracy of 85.45%, recall of 85.62% and precision of 85.63% while keeping the amount of features considerably low with only 99 features. When compared to methods of other contexts of this document, it is possible to notice that the Hybrid methods presented an overall better performance. This result shows that utilizing concepts from diverse fields and extracting image information using various methods can lead to improved performance in the texture classification field.

With a more detailed examination, we can analyse the proposed method's behavior by comparing different combinations of the final vector features. If we take the combined vector and evaluate accuracy of the three descriptors individually, we see that Local Binary Pattern gets 51.01%, Haralick gets 79.32% and the proposed Closeness method alone ($\mathcal{H}_G$) performs better with 80.34%. Pairing up the features we get that Closeness + LBP and Closeness + Haralick perform similarly with accuracy of 83.74% and 83.26%, respectively, while LBP + Haralick ($\mathcal{H}_E$) underperforms with accuracy of 80.24%. However, if we analyse the algorithm with the feature vector combined of the three descriptors, we get an accuracy of 85.45%. That's an increase of more than 1.7% in the classification performance. These results can be seen on Table 5.7.

On Figure 5.2 the performance of all the methods can be seen. The Proposed method came in first, followed by the PCANet, which are the Hybrid ones. Tied in third place were the T-CNN-3 and the Haralick and, in fourth, the Diffusion in Networks method. It is cool to notice that Haralick is a very classic and old method but can still compete with

Variations	**Features**	**Acc.**	**Rec.**	**Prec.**
Closeness	75	80.34%	80.61%	81.14%
LBP	10	51.01%	51.48%	51.56%
Haralick	14	79.32%	79.63%	79.65%
LBP + Haralick	24	80.24%	80.51%	80.41%
Closeness + LBP	85	83.74%	83.93%	84.15%
Closeness + Haralick	89	83.26%	83.44%	83.72%
Proposed Method	99	85.45%	85.62%	85.63%

Acc=Accuracy; Rec=Recall; Prec=Precision.

Table 5.7: Combined feature vector variations metrics.

some state-of-the-art methods that use deep learning or graph theory.

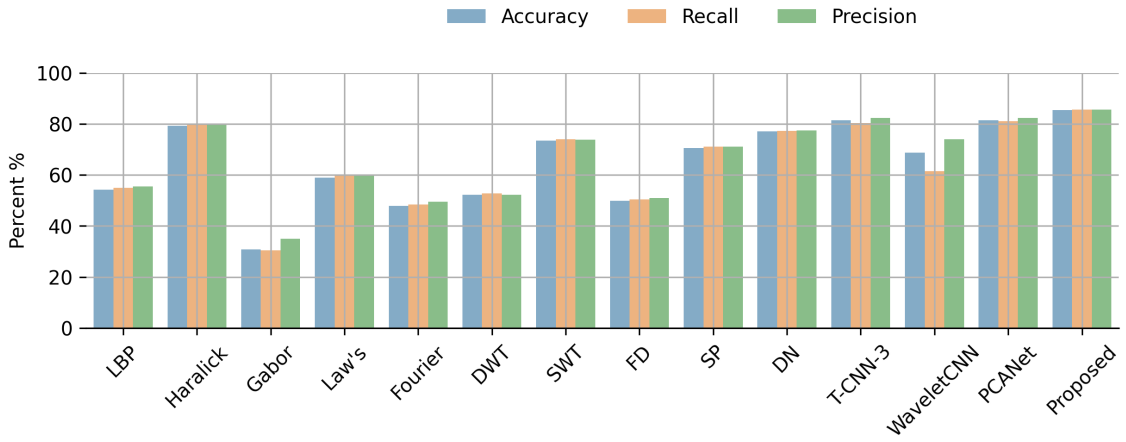


Figure 5.2: Bar plot with the accuracy, recall and precision of the methods analysed: LBP (sub-section 4.1.2), Haralick (sub-section 4.1.1), Gabor (sub-section 4.2.2), Law's (sub-section 4.2.1), Fourier (sub-section 4.2.3), DWT and SWT (sub-section 4.2.4), FD - Fractal Dimension (sub-section 4.2.5), SP - Shortest Paths (sub-section 4.3.1), DN - Diffusion in Networks (sub-section 4.3.2), T-CNN-3 (sub-section 4.4.1), WaveletCNN (sub-section 4.4.2), PCANet (sub-section 4.5.1), Proposed Method (sub-section 4.5.2).

When analysing the performance of each method for every class individually, the Table 5.8 can be formed. For each class, the precision and recall of each method were measured. The highest results for each class were styled in bold. The proposed method presented great performance on every class with best result for both precision and recall for EA and GGO, best precision for healthy and ST and best recall for PC and HC. PCANet presented best precision for PC and best recall for ST. SWT presented best precision for HC.

Furthermore, it is important to analyse the method's performance on the PC and GGO classes once they are standard features of COVID-19 patients [He et al., 2020]. The highest scores on these classes belongs to the Proposed method with 92.39% and 88.98% for precision and recall of GGO, and 86.54% and 89.56% of precision and recall

on PC. Its performance on the Healthy class is also great with precision and recall of 88.35% and 86.54%, respectively.

Methods	Healthy		PC*		EA		ST		HC		GGO*	
	Prec.	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**
LBP	65.29%	57.6%	60.77%	71.94%	44.8%	55.0%	34.11%	36.16%	60.23%	55.2%	67.37%	53.89%
Haralick	81.51%	73.33%	77.89%	88.15%	70.58%	77.95%	73.73%	67.02%	78.60%	81.73%	88.81%	85.79%
HOG	25.52%	7.01%	36.24%	1.88%	47.79%	3.1%	36.56%	16.12%	22.38%	33.76%	21.87%	72.78%
Law	52.07%	49.22%	72.28%	82.11%	43.06%	47.89%	41.24%	41.92%	78.3%	74.57%	71.64%	63.21%
Gabor	40.85%	53.65%	36.65%	41.34%	32.33%	46.09%	34.74%	27.37%	41.59%	35.7%	45.72%	26.05%
Fourier	39.25%	33.62%	49.17%	56.61%	38.99%	45.85%	35.37%	39.98%	74.84%	60.53%	58.41%	53.73%
DWT	51.52%	41.06%	53.22%	66.2%	39.45%	42.8%	38.06%	36.01%	63.01%	64.15%	67.39%	66.12%
SWT	71.99%	64.48%	74.91%	86.3%	58.17%	71.56%	65.92%	56.29%	**86.78%**	82.93%	84.66%	82.79%
Fractal Dimension	52.46%	47.09%	67.24%	72.44%	32.6%	36.32%	37.13%	41.26%	53.13%	50.18%	61.53%	54.47%
Shortest Paths	77.35%	70.34%	74.52%	78.86%	76.91%	82.36%	52.91%	54.19%	65.35%	63.02%	77.63%	77.13%
Diffusion in Networks	76.4%	80.99%	77.82%	78.67%	82.51%	87.08%	64.26%	63.15%	82.55%	73.77%	79.85%	79.62%
T-CNN-3	76.14%	**88.84%**	80.77%	74.63%	81.05%	74.85%	69.63%	64.01%	75.68%	78.97%	78.00%	70.58%
WaveletCNN	60.35%	71.13%	75.26%	74.25%	72.57%	69.76%	57.25%	47.31%	76.20%	68.81%	76.46%	73.32%
PCANet	81.59%	87.53%	**88.17%**	73.30%	81.07%	82.36%	72.13%	**76.15%**	83.94%	77.19%	82.39%	87.20%
Proposed Method	**88.35%**	86.54%	86.54%	**89.56%**	**85.58%**	**90.72%**	**76.5%**	74.32%	82.96%	**83.72%**	**92.39%**	**88.98%**

Table 5.8: Precision and Recall for all analysed methods for each class in the dataset: PC is Pulmonary Consolidation, EA is Emphysematous Area, ST is Septal Thickening, HC is Honeycomb and GGO is Ground-glass opacity. Classes with (*) are classes presented by COVID-19 patients.

Chapter 6

Conclusion and Future Works

In this work, an analysis of various methods used to characterize images was performed over a dataset of regions of interest of HRCT scans of the lungs and, with this study, a new approach that presents better performance on performing classification on the dataset is proposed. The dataset is composed of 6 classes: healthy, pulmonary consolidation, emphysematous area, septal thickening, honeycomb and ground-glass opacity. Each of these classes are textural patterns presented by some Diffuse Parenchymal Lung Diseases (DPLDs). This family of diseases are very difficult to diagnose because they can present visual similarities to each other. The objective of this work is to explore the capability of the methods on recognizing these patterns and to suggest a new approach that enhances the outcomes in addressing this issue.

The proposed method combines graph-based and statistical texture descriptors (Haralick and LBP) to classify the images into the 6 mentioned classes. The original image is mapped into directed complex networks. Each pixel was considered as a vertex of the network connecting vertices within a given radius. The closeness centrality metric is extracted from multiple scaled networks, and a vector of texture features is formed from the sum of the closeness values within a specific pixel intensity interval. The final feature vector is composed of the vector generated by this process, the 14 haralick features and a histogram of the resultant image from the LBP operator.

After the experiments, it is possible to notice that the Proposed method had the best overall results on classifying DPLDs using this dataset. Followed by the PCANet – a network architecture method that uses concepts of Principal Component Analysis to generate filter for the images – and, T-CNN-3 – a member of a family of Convolution Neural Networks for Texture Classification.

Another important point is to analyze the method's metrics for each class. The proposed method presented the highest scores for most of the classes. Furthermore, it is possible to analyse the method's performance as an aid on the problem of diagnosing patients with COVID-19, once its patients present the pulmonary consolidation and ground-glass opacity. The proposed approach presented a great performance on these classes.

Further studies on the method can be performed to improve its performance on this application. Given that this approach combines various methods and concepts, there is room for future research to experiment with alternative combinations in order to enhance effectiveness. In addition to the methods outlined in this study, other works have explored the field of Information Theory concerning the classification of textures. The next steps include the investigation of various domains to construct an efficient texture classification method, leveraging descriptors that can accurately distinguish between different patterns.

Bibliography

[Abusham and Bashir, 2011] Abusham, E. E. and Bashir, H. K. (2011). Face recognition using local graph structure (lgs). page 169–175.

[Albers and Alexanderson, 2008] Albers, D. and Alexanderson, G. L. (2008). *Mathematical People*.

[Andrearczyk and Whelan, 2016] Andrearczyk, V. and Whelan, P. F. (2016). Using filter banks in convolutional neural networks for texture classification. *Pattern Recognition Letters*, 84:63–69.

[Avelar et al., 2020] Avelar, P. H., Tavares, A. R., da Silveira, T. L., Jung, C. R., and Lamb, L. C. (2020). Superpixel image classification with graph attention networks. *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*.

[Cabral et al., 2014] Cabral, R. S., Frery, A. C., and Ramírez, J. A. (2014). Variability analysis of complex networks measures based on stochastic distances. *Physica A: Statistical Mechanics and its Applications*, 415:73–86.

[Chan et al., 2015] Chan, T.-H., Jia, K., Gao, S., Lu, J., Zeng, Z., and Ma, Y. (2015). Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, 24(12):5017–5032.

[Comin et al., 2012] Comin, C., Viana, M., Antiqueira, L., and da F. Costa, L. (2012). Diffusion in directed modular networks.

[Couto et al., 2017] Couto, L. N., Backes, A. R., and Barcelos, C. A. (2017). Texture characterization via deterministic walks' direction histogram applied to a complex network-based image transformation. *Pattern Recognition Letters*, 97:77–83.

[da Mata, 2020] da Mata, A. S. (2020). Complex Networks: a Mini-review. *Brazilian Journal of Physics*, 50(5):658–672.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1.

[Darapureddy et al., 2021] Darapureddy, N., Karatapu, N., and Battula, T. K. (2021). Comparative Analysis of Texture Patterns on Mammograms for Classification. *Traitement du Signal*, 38(2):379–386.

[de Mesquita Sa et al., 2013] de Mesquita Sa, J. J. J., Backes, A. R., and Cortez, P. C. (2013). Texture analysis and classification using shortest paths in graphs. *Pattern Recognition Letters*, 34(11):1314–1319.

[Fan et al., 2018] Fan, T.-w., Malhi, H., Varghese, B., Cen, S., Hwang, D., Aron, M., Rajarubendra, N., Desai, M., and Duddalwar, V. (2018). Computed tomography-based texture analysis of bladder cancer: Differentiating urothelial carcinoma from micropapillary carcinoma. *Abdominal Radiology*, 44(1):201–208.

[Fernandez et al., 2011] Fernandez, A., Alvarez, M. X., and Bianconi, F. (2011). Image classification with binary gradient contours. *Optics and Lasers in Engineering*, 49(9-10):1177–1184.

[Fujieda et al., 2018] Fujieda, S., Takayama, K., and Hachisuka, T. (2018). Wavelet convolutional neural networks. *CoRR*, abs/1805.08620.

[Ghalati et al., 2022] Ghalati, M. K., Nunes, A., Ferreira, H., Serranho, P., and Bernardes, R. (2022). Texture analysis and its applications in biomedical imaging: A survey. *IEEE Reviews in Biomedical Engineering*, 15:222–246.

[Giakoumoglou, 2021] Giakoumoglou, N. (2021). Pyfeats: Open-source software for image feature extraction. https://github.com/giakou4/pyfeats.

[Gonçalves et al., 2016] Gonçalves, W. N., da Silva, N. R., da Fontoura Costa, L., and Bruno, O. M. (2016). Texture recognition based on diffusion in networks. *Inf. Sci.*, 364(C):51–71.

[Gonzalez and Woods, 2018] Gonzalez, R. C. and Woods, R. E. (2018). *Digital Image Processing*. Pearson, 4 edition.

[Haralick et al., 1973] Haralick, R. M., Shanmugam, K., and Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621.

[He et al., 2020] He, X., Zheng, J., Ren, J. L., Zheng, G., and Liu, L. (2020). Chest high-resolution computed tomography imaging findings of coronavirus disease 2019 (Covid-19) pneumonia. *International Journal of Radiation Research*, 18(2):343–349.

[Humeau-Heurtier, 2019] Humeau-Heurtier, A. (2019). Texture Feature Extraction Methods: A Survey. *IEEE ACCESS*, 7:8975–9000.

[Jollife, 2010] Jollife, I. T. (2010). *Principal component analysis*. Springer.

[Laws, 1980] Laws, K. I. (1980). Rapid Texture Identification. In Wiener, T. F., editor, *Image Processing for Missile Guidance*, volume 0238, pages 376 – 381. International Society for Optics and Photonics, SPIE.

[Lingley-Papadopoulos et al., 2008] Lingley-Papadopoulos, C., Loew, M., Manyak, M., and Zara, J. (2008). Computer recognition of cancer in the urinary bladder using optical coherence tomography and texture analysis. *Journal of biomedical optics*, 13:024003.

[Lu et al., 2018] Lu, R.-S., Dennison, E., Denison, H., Cooper, C., Taylor, M., and Bottema, M. J. (2018). Texture analysis based on gabor filters improves the estimate of bone fracture risk from dxa images. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 6(4):453–464.

[Lü et al., 2016] Lü, L., Chen, D., Ren, X.-L., Zhang, Q.-M., Zhang, Y.-C., and Zhou, T. (2016). Vital nodes identification in complex networks. *Physics Reports*, 650:1–63. Vital nodes identification in complex networks.

[M. et al., 2017] M., H., S., A., Mohammed, A., and Jennane, R. (2017). Histogram of oriented gradients and texture features for bone texture characterization. *International Journal of Computer Applications*, 165(3):23–28.

[Mattonen et al., 2014] Mattonen, S. A., Palma, D. A., Haasbeek, C. J., Senan, S., and Ward, A. D. (2014). Early prediction of tumor recurrence based on ct texture changes after stereotactic ablative radiotherapy (sabr) for lung cancer. *Medical Physics*, 41(3).

[Monemian and Rabbani, 2019] Monemian, M. and Rabbani, H. (2019). A New Texture-Based Segmentation Method fc r Optical Coherence Tomography Images. In *2019 41ST Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE Engineering in Medicine and Biology Society Conference Proceedings, pages 4750–4753. 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Berlin, GERMANY, JUL 23-27, 2019.

[Mori, 2020] Mori, K. (2020). Chapter 4 - cad in lung. In Zhou, S. K., Rueckert, D., and Fichtinger, G., editors, *Handbook of Medical Image Computing and Computer Assisted Intervention*, The Elsevier and MICCAI Society Book Series, pages 91–107. Academic Press.

[Nguyen et al., 2016] Nguyen, T. P., Vu, N.-S., and Manzanera, A. (2016). Statistical binary patterns for rotational invariant texture classification. *Neurocomputing*, 173:1565–1577.

[Noh and Rieger, 2004] Noh, J. D. and Rieger, H. (2004). Random walks on complex networks. *Phys. Rev. Lett.*, 92:118701.

[Ojala et al., 1996] Ojala, T., Pietikainen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29(1):51–59.

[Ojala et al., 2002] Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987.

[Pereyra et al., 2014] Pereyra, L. C., Rangayyan, R. M., Ponciano-Silva, M., and Azevedo-Marques, P. M. (2014). Fractal analysis for computer-aided diagnosis of diffuse pulmonary diseases in HRCT images. *IEEE MeMeA 2014 - IEEE International Symposium on Medical Measurements and Applications, Proceedings*.

[Raupov et al., 2016] Raupov, D. S., Myakinin, O. O., Bratchenko, I. A., Zakharov, V. P., and Khramov, A. G. (2016). Skin cancer texture analysis of oct images based on haralick, fractal dimension, markov random field features, and the complex directional field features. *Optics in Health Care and Biomedical Optics VII*.

[Sharma and Ghosh, 2015] Sharma, M. and Ghosh, H. (2015). Histogram of gradient magnitudes: A rotation invariant texture-descriptor. In *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE International Conference on Image Processing ICIP, pages 4614–4618. Inst Elect & Elect Engineers; IEEE Signal Proc Soc. IEEE International Conference on Image Processing (ICIP), Quebec City, CANADA, SEP 27-30, 2015.

[Song et al., 2012] Song, L., Liu, X., Ma, L., Zhou, C., Zhao, X., and Zhao, Y. (2012). Using hog-lbp features and mmp learning to recognize imaging signs of lung lesions. *2012 25th IEEE International Symposium on Computer-Based Medical Systems (CBMS)*.

[Sørensen et al., 2008] Sørensen, L., Shaker, S. B., and de Bruijne, M. (2008). Texture classification in lung ct using local binary patterns. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, page 934–941.

[Tsiaparas et al., 2011] Tsiaparas, N., Golemati, S., Andreadis, I., Stoitsis, J. S., Valavanis, I., and Nikita, K. S. (2011). Comparison of multiresolution features for texture classification of carotid atherosclerosis from b-mode ultrasound. *IEEE Transactions on Information Technology in Biomedicine*, 15(1):130–137.

[Tuzuner et al., 2020] Tuzuner, A. B., Erogul, O., Atac, G. K., and Er, H. C. (2020). Classification of Ultrasonographic Thyroid Tumor Images to TIRADS Categories via

Texture Analysis Methods. In *2020 Medical Technologies Congress (TIPTEKNO)*. Biyomedikal ve Klinik Muhendisligi Dernegi; Izmir Ekonomi Univ; Izmir Katip Celebi Univ. 2020 Medical Technologies Congress (TIPTEKNO), ELECTR NETWORK, NOV 19-20, 2020.

[Vince et al., 2000] Vince, D., Dixon, K., Cothren, R., and Cornhill, J. (2000). Comparison of texture analysis methods for the characterization of coronary plaques in intravascular ultrasound images. *Computerized Medical Imaging and Graphics*, 24(4):221–229.

[Wei et al., 2018] Wei, G., Cao, H., Ma, H., Qi, S., Qian, W., and Ma, Z. (2018). Content-based image retrieval for Lung Nodule Classification Using Texture Features and Learned Distance Metric. *Journal of Medical Systems*, 42(1).

[Wu et al., 1992] Wu, C.-M., Chen, Y.-C., and Hsieh, K.-S. (1992). Texture features for classification of ultrasonic liver images. *IEEE Transactions on Medical Imaging*, 11(2):141–152.

[Young et al., 1981] Young, I., Hall, A., Pallis, C., Bydder, G., Legg, N., and Steiner, R. (1981). Nuclear magnetic resonance imaging of the brain in multiple sclerosis. *The Lancet*, 318(8255):1063–1066.

[Zhang et al., 2015] Zhang, J., Liang, J., Zhang, C., and Zhao, H. (2015). Scale invariant texture representation based on frequency decomposition and gradient orientation. *Pattern Recognition Letters*, 51:57 – 62.

[Zhang et al., 2008] Zhang, J., Tong, L., Wang, L., and Li, N. (2008). Texture analysis of multiple sclerosis: A comparative study. *Magnetic Resonance Imaging*, 26(8):1160–1166.

[Zhao et al., 2018] Zhao, Z., Yang, G., Lin, Y., Pang, H., and Wang, M. (2018). Automated glioma detection and segmentation using graphical models. *PLOS ONE*, 13(8).

[Álvaro Albuquerque et al., 2022] Álvaro Albuquerque, Mendes, Y., Almeida, E., Cabral, R., and Queiroz, F. (2022). Combining statistical and graph-based approaches to classification of interstitial pulmonary diseases. In *Anais Estendidos do XXXV Conference on Graphics, Patterns and Images*, pages 119–123, Porto Alegre, RS, Brasil. SBC.