Pedro Henrique Bandeira Diniz

Detection of Regions of White Matter Lesions of the Brain in T1 and Flair Images

Tese de Doutorado

Thesis presented to the Programa de Pós-Graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências - Informática

Advisor: Prof. Marcelo Gattass
Co-Advisor: Prof. Aristófanes Corrêa Silva

Rio de Janeiro
May 2018

Pontifícia Universidade Católica do Rio de Janeiro

Pedro Henrique Bandeira Diniz

Detection of Regions of White Matter Lesions of the Brain in T1 and Flair Images

Thesis presented to the Programa de Pós-Graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências – Informática. Approved by the undersigned Examination Committee.

Rio de Janeiro, May 8th, 2018

Pedro Henrique Bandeira Diniz

Graduated in Computer Science from Universidade Federal do Maranhão – UFMA in 2012, he obtained the degree of Mestre at Universidade Federal do Maranhão – UFMA in Electrical Engineering in 2014

Bibliographic data

Diniz, Pedro Henrique Bandeira

Detection of regions of white matter lesions of the brain in T1 and Flair images / Pedro Henrique Bandeira Diniz; advisor: Marcelo Gattass; co-advisor: Aristófanes Corrêa Silva. – Rio de Janeiro, Departamento de Informática, 2018.

98 f.: il. color. ; 30 cm

1. Tese (doutorado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui referências bibliográficas.

1. Informática – Teses. 2. Lesão na substância branca. 3. Ressonância magnética. 4. Detecção auxiliada por computador. 5. SLIC0. 6. Redes neurais convolucionais. I. Gattass, Marcelo. II. Silva, Aristófanes Corrêa. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD:004

Acknowledgments

Firstly, I thank God for everything in my life.

I would like to thank my advisor Prof. Marcelo Gattass and co-advisor Prof. Aristófanes for the support of my Ph.D study and related research.

Besides my advisor and co-advisor, I would like to thank the rest of my thesis committee: Prof. Waldemar Celes Filho, Prof. Melissa Lemos Cavaliere, Prof. Paulo Cezar Pinto Carvalho, Prof. Raul Queiroz Feitosa, and Prof. Nina Ventura Wilner, for their interest and insightful comments, but also for the hard question which incented me to widen my research.

I thank my brother João and my friends Thales and Eduardo for the discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years.

Last but not the least, I would like to thank my family: my parents and to my brother and sister for supporting me throughout writing this thesis and my life in general.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

Abstract

Diniz, Pedro Henrique Bandeira; Gattass, Marcelo (Advisor). **Detection of Regions of White Matter Lesions of the Brain in T1 and Flair Images.** Rio de Janeiro, 2018. 98p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

White matter lesions are non-static brain lesions that have a prevalence rate up to 98% in the elder population, although it is also present in the young population. Because it may be associated with several brain diseases, it is important to detect them as early as possible. Magnetic resonance imaging provides three-dimensional data for visualization and analysis of soft tissues as it contains rich information about their anatomy. However, the amount of data acquired for these images may be too much for manual analysis/interpretation alone, representing a difficult and time-consuming task for specialists. Therefore, this doctoral thesis presents four new computational methods to automatically detect white matter lesions in magnetic resonance images, based mainly on algorithms SLIC0 and Convolutional Neural Networks. Our primary objective is to provide the necessary tools for specialists to accelerate their works and suggest a second opinion. From the four proposed methods, the one that achieved best results was applied on 91 magnetic resonance images, and achieved an accuracy of 97.93%, specificity of 98,02% and sensitivity of 90,12%, without using any candidate reduction techniques.

Keywords

White matter lesion; Magnetic resonance imaging; Computer-aided detection; SLIC0; Convolutional neural networks.

Resumo

Diniz, Pedro Henrique Bandeira; Gattass, Marcelo (Orientador). **Detecção de Regiões de Lesões na Substância Branca do Cérebro em Imagens T1 e FLAIR.** Rio de Janeiro, 2018. 98p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

As lesões da substância branca são lesões cerebrais não estáticas que têm uma taxa de prevalência de até 98% na população idosa, embora também esteja presente na população jovem. Uma vez que elas podem estar associadas a várias doenças cerebrais, é importante detectá-las o mais cedo possível. A ressonância magnética fornece dados tridimensionais para visualização e análise de tecidos moles, pois contém informações ricas sobre sua anatomia. No entanto, a quantidade de dados adquiridos para essas imagens pode ser excessiva para análise / interpretação manual, representando uma tarefa difícil e demorada para especialistas. Portanto, esta tese de doutorado apresenta quatro novos métodos computacionais para detectar automaticamente lesões de substância branca em imagens de ressonância magnética, baseadas principalmente nos algoritmos SLIC0 e Convolutional Neural Networks. Nosso principal objetivo é fornecer as ferramentas necessárias para que os especialistas acelerem seus trabalhos e sugiram uma segunda opinião. Dos quatro métodos propostos, o que obteve melhores resultados foi aplicado em 91 imagens de ressonância magnética, e obteve uma precisão de 97,93%, especificidade de 98,02% e sensibilidade de 90,12%, sem utilizar nenhuma técnica de redução de candidatos.

Palavras-chave

Lesão na substância branca; Ressonância magnética; Detecção auxiliada por computador; SLIC0; Redes neurais convolucionais.

Table of contents

1. Introduction 14
1.1. Motivation 17
1.2. Objectives 18
1.2.1. General Objective 18
1.2.2. Specific Objectives 18
1.3. Work Organization 18
2. Theoretical Foundation 19
2.1. DICOM Standard 19
2.2. Clustering 20
2.2.1. Superpixels 20
2.2.1.1. SLIC 21
2.2.1.2. SLIC0 21
2.3. Image registration 23
2.3.1. Rigid registration 24
2.3.1.1. Image Transformation Function 24
2.3.1.2. Image Interpolator 24
2.3.1.3. Images Metric of Similarity 25
2.3.1.4. Registration Optimizer 25
2.4. Artificial Neural Networks (ANNs) 26
2.4.1. Artificial neuron 26
2.4.2. Multilayer Perceptron (MLP) 27
2.4.3. Deep Learning 28
2.4.4. Overfitting and Dropout 29
2.4.5. Convolutional Neural Networks (CNN) 30
2.4.5.1. Convolution Layer 31
2.4.5.2. Downsampling layer 32
2.4.5.3. Fully connected layers 33
2.5. Validation Metrics 33
3. Methods 35
3.1. Method 1 35
3.1.1. Image Acquisition 35
3.1.2. Image Preprocessing 36
3.1.2.1. Skull Stripping 36
3.1.2.2. Histogram Matching 37
3.1.3. Candidates Segmentation 37
3.1.4. Candidates Classification 38
3.1.4.1. Generation of Patches 38
3.1.4.2. CNN Train and Test 40

3.2. Method 2 ... 42
3.2.1. Candidates Classification ... 42
3.2.1.1. Generation of Multiscale Patches ... 42
3.2.1.2. Parallel CNN Train and Test ... 43
3.3. Method 3 ... 45
3.3.1. Image Acquisition ... 45
3.3.2. Image Preprocessing ... 46
3.3.2.1. Skull Stripping ... 47
3.3.2.2. Spacing and Plane Correction ... 47
3.3.2.3. Image Alignment ... 47
3.3.2.4. Histogram Matching ... 48
3.3.3. Candidates Segmentation ... 49
3.3.4. Candidates Classification ... 49
3.3.4.1. Generation of Multimodal Patches ... 49
3.3.4.2. Parallel CNN Train and Test ... 50
3.4. Method 4 ... 51
3.4.1. Candidates Classification ... 51
3.4.1.1. Generation of Multimodal Patches ... 52
3.4.1.2. CNN Train and Test ... 54
4. Results and discussion ... 56
4.1. Train, validation and test images ... 56
4.2. Training step ... 59
4.3. Test step ... 60
4.4. Discussion ... 60
4.5. Study of cases ... 62
4.6. Comparison with related works ... 64
5. Conclusions ... 67
5.1. Work Overview ... 67
5.2. Work Evaluation ... 67
5.3. Work Contributions ... 68
5.4. Future Works ... 68
References ... 70
A. WML detection software ... 84
A.1. Requirements and Specifications ... 84
A.2. Use Cases ... 85
A.3. Development ... 89
A.3.1. Platforms and Technologies ... 89
A.3.2. Modeling and Architecture ... 89
A.4. Software ... 93
A.4.1. Graphical User Interface ... 93
A.4.2. Features and How to Use ... 94

List of figures

Figure 1: SLIC versus SLIC0. 22
Figure 2: Activation functions. 27
Figure 3: Illustration of the use of dropout in a neural network. 30
Figure 4: Example CNN Architecture. 31
Figure 5: Convolution layer illustration. 32
Figure 6: Illustration of the subsampling layer. 32
Figure 7: Method using only FLAIR MRI. 35
Figure 8: Brain segmentation in FLAIR MRI. 37
Figure 9: Histogram Matching. 37
Figure 10: SLIC0 applied in FLAIR MRI. 38
Figure 11: Generation of CNN patches. 39
Figure 12: Selecting WML superpixels. 40
Figure 13: Architecture of the CNN of the first method. 41
Figure 14: Method using only FLAIR MRI. 42
Figure 15: Generation of multiscale patches. 43
Figure 16: Architecture of the parallel CNN with multiscale patches. 45
Figure 17: Method using T1 and FLAIR MRI. 46
Figure 18: Brain segmentation. 47
Figure 19: Resampling T1 MRI to axial plane. 48
Figure 20: Cropping images to their segment brain 3D bounding box. 48
Figure 21: Aligning T2 and T1 images using 3D rigid registration. (a) T1 slice.. 49
Figure 22: Generation of multimodal patches. 50
Figure 23: Architecture of the parallel CNN with multimodal patches. 51
Figure 24: Method using T1 and FLAIR MRI. 52
Figure 25: Generation of comparing patches. 54
Figure 26: Architecture of CNNs with comparing patches. 55
Figure 27: Study of case 1. 63
Figure 28: Study of case 2. 63
Figure 29: Study of case 3. 64
Figure 30: Study of case 4. 64

Figure 31: Use Cases Diagram of the WML segmentation software. 86
Figure 32: Model-View-Presenter standard. 90
Figure 33: Class Diagram. 91
Figure 34: Sequence Diagram of DICOM volume reading. 92
Figure 35: Sequence Diagram of automatic WML segmentation. 93
Figure 36: GUI Components. 94
Figure 37: DICOM directory selection window. 95
Figure 38: Displaying volume slice. 95
Figure 39: Applying zoom and pan on volume slice. 96
Figure 40: Contrast Window Adjustment. 96
Figure 41: Manual segmentation process. 97
Figure 42: Manual segmentation process. 98

List of tables

Table 1: Number of WML and regions of WML per patient in database 1.57
Table 2: Number of WML and regions of WML per patient in database 2.58
Table 3: Characteristics of the groups of images of the training step in database 1.. 58
Table 4: Characteristics of the groups of images of the training step in database 2.. 59
Table 5: Characteristics of the groups of images of the test step in database 1.. 59
Table 6: Characteristics of the groups of images of the test step in database 2.. 59
Table 7: Validation metrics of the training step. 60
Table 8: Validation metrics of the test step. 60
Table 9: Comparison with related works 65
Table 7: Summary of platforms and technologies used to develop the software. 89

List of acronyms

WML	White Matter Lesion
CT	Computerized Tomography
MRI	Magnetic Resonance Imaging
FLAIR	Fluid Attenuated Inversion Recovery
CAD	Computer-Aided Detection
CADx	Computer-Aided Diagnostic
SLIC	Simple Linear Iterative Clustering
SLIC0	Simple Linear Iterative Clustering Zero Parameter
MLP	Multilayer Perceptron
CNN	Convolutional Neural Network
SVM	Support Vector Machines
DICOM	Digital Imaging and Communications in Medicine
ACR	American College of Radiology
PACS	Picture Archiving and Communications Systems
NEMA	National Electrical Manufactures Association
MVP	Model-View-Presenter
GUI	Graphical User Interface

behold, I have done according to your words. Behold, I have given you a wise and discerning heart, so that there has been no one like you before you, nor shall one like you arise after you.

1 Kings 3:12

1. Introduction

White matter comprises over half the human brain and is located underlying the gray matter cortex. It is composed of neuronal fibers coated with electrical insulation called myelin [4]. Individual differences in cognitive development are correlated with differences in white matter structure in specific brain regions [5–7]. However, the process of myelination is adjustable because of the interaction and experience of the individual on the environment, and it affects information processing by regulating the velocity and sync of impulse conduction between distant cortical regions. The learning process of the skills, such as play an instrument, are accompanied by increased organization of white matter structure in appropriate brain tracts involved in musical performance [8]. In other words, the level of white matter structure increases proportionately to the practice related a one skill, indicating white matter changes in acquiring the skill rather than performance being predetermined by a limitation on white matter development.

A wide range of psychiatric and neurological disorders result from damage or disease affecting the myelin sheath on nerve fibers, changing the white matter. Pathologically, these white matter changes are characterized by partial loss of myelin, axons, and oligodendroglial cells [3]. Abnormal changes in brain white matter are frequently named in the literature as white matter lesions (WML) [1, 9, 10], white matter hyperintensity [1, 2, 9, 11, 12] or leukoaraiosis [2, 10, 11, 13]. However, WML are non-static lesions. The lesions may progress, or even regress, over time. Several longitudinal studies have investigated the rate and predictors for progression of WML [14–18].

WML may be associated with several factors, and the affected population rate may vary according to the region, anomaly, and population sample to which the study focuses. However, it is known that they are often detected in elderly people [19–24]. For example, studies indicate that in populations over 65 years old, the incidence of WML is higher than 30% and increases with age [22, 25] and other studies point to a prevalence rate in the range of 15% to 98% in the elderly population [20–23, 26–30].

Because WML may be associated with several factors, there are a wide range of causes and consequences associated with them. As risk factors, they may be related to hypertension [21, 31–37], diabetes mellitus [38], hyperlipidemia [7], smoking [39], small vessel disease [40, 41], multiple sclerosis [42], Parkinsonism [43] (about 30-55% of patients [44–46]), several stroke types [47, 48] (about 67-98% [49–52]), Alzheimer’s disease [53] (about 28.9% to 100% [54–56]), dementia [24, 41, 56], higher blood pressure and fluctuations blood pressure [14, 31, 37, 57], atherosclerosis [25] and heredity [58].

In past times, WML were observed on Computerized Tomography (CT) when only CT imaging was widely available [48]. However, nowadays, the presence of white matter abnormalities is usually established with Magnetic Resonance Imaging (MRI). It has several advantages over other imaging techniques because it provides three-dimensional data with the possibility to detect and emphasize contrast differences in soft tissues by manipulating its parameters, providing rich information about the human soft tissue anatomy [59–62]. Although different MRI modalities exist, the literature point Fluid Attenuated Inversion Recovery (FLAIR) as probably the best MRI technique to observe WML [24, 63, 64]. WML detected on MRI of modality FLAIR are observed as high value signals or hyperintensity regions [24, 48, 64], contrasting with hypointense healthy tissue, but WML may be confused with some healthy tissues that also presents hyperintensity. Thus, it is not uncommon to use MRI FLAIR combined with other MRI techniques such as T1, T2 and Proton Density-Weighted [65]. In the T1 case, WML regions hyperintense in FLAIR may appear as iso or hypointense in T1, so their information may be combined with FLAIR [65, 66].

In order to support specialists in the identification of anomalies in human body, many researches have invested substantially in the development of computational systems. These systems are named Computer-Aided Detection (CAD) and Computer-Aided Diagnostic (CADx). They are usually a product of expert knowledge combination which includes image processing, computational intelligence, medical physics and medicine - a truly interdisciplinary research field. The goal of these systems is improving specialist performance, increasing efficiency in medical data analysis by providing additional information and judgment about them and reducing workload. So, it is expected specialists to be more productive in the process of detection and diagnosing abnormalities, doing that earlier and faster because they will have more time to think about the interpretation of their findings instead of focusing on all regions of the images. In the literature, many CAD examples are applied in several medical images types to provide support in the diseases analysis can be found [68–72].

Detection algorithms in CAD systems are usually divided in the following steps (sometimes it is not required all of them): (1) image acquisition, (2) pre-processing, (3) candidates segmentation, (4) features extraction (sometimes followed by features selection) and classification. Because of the complexity of performing all these steps, it is not difficult to find researches dividing CAD system into two studies or main parts: (A) regions of interest (ROI) segmentation or candidates segmentation and (B) candidates classification [68–72]. The purpose of the candidates segmentation is to segment as much suspect regions as possible. This may imply in many candidates and the presence of many false positive ones. So, the purpose of the candidates classification is to reduce these false positive candidates. This work presents both candidates segmentation and classification.

Several researches related to the task of candidates segmentation in brain MRI were proposed in literature. Some of them are based on relatively simple methods such as local threshold [73], the intensity gradient [74], fuzzy connectedness [75], region growing [76, 77], mathematical morphology [78], adaptive thresholding [79] and gaussian mixture model [80].

Recently, superpixels-based segmentations have been widely used, including in brain MRI [81, 82]. They take into consideration that neighboring pixels are usually highly correlated and seeks to group them into perceptually meaningful atomic regions. One of these segmentations is the Simple Linear Iterative Clustering (SLIC), which uses a weighted distance measure that combines color and spatial proximity, while simultaneously providing control over the size and compactness of the superpixels [83]. However, if the image is smooth in certain regions but highly textured in others, SLIC produces smooth regular-sized superpixels in the smooth regions and highly irregular superpixels in the textured regions [84]. Therefore, this thesis uses an improved version of SLIC, SLIC Zero Parameter (SLIC0) [83], which adaptively chooses the compactness factor for each superpixel individually and generates regular shaped superpixels in both textured and non-textured regions alike, with high computational efficiency.

A golden rule in computing, the Moore Law [114], describes the exponential improvement of computer hardware. According to it, the number of transistors incorporated in a chip will approximately double every 24 months. Nowadays, after more than 50 years, this rule is still determining the rhythm of innovation and development, bringing several benefits impacting in the economic, technological and social areas. Accompanying this evolution of computational power, more robust software has been developed in the various areas of human knowledge. Theories previously computationally impractical and others not even imagined 50 years ago, are possible today. More specifically in machine learning, deep learning techniques have been developed and applied to the most diverse types of data in the search for patterns that discriminate one or more classes of states, objects or events. It is extremely computationally expensive to train, directly proportional to the complexity of the network and the dimensionality of the data. In addition, it requires a large amount of data. In recent years, these techniques have been applied rather using GPU. According to Schmidhuber [85], GPU-based computers 2 years ago have a million times the computational power of desktop machines of the early 1990s.

Convolutional neural network (CNN) [89] is one particular type of deep learning that have recently achieved the best results in image classification challenges. It has provided excellent results applied in different computational vision tasks as traffic sign recognition [93], segmentation of biological images [94], detection of pedestrians [95] and human bodies [96] and detection and recognition of faces [97, 98] and text [99] in natural images. In recent years, CNN have also been applied in the field of medical image [88, 100–102], including brain MRI analysis [67, 103–106].

In this thesis, we present the combination of two state-of-arts techniques: (1) SLIC0 algorithm for WML and non-WML regions segmentation and (2) CNN for classification of these regions.

1.1. Motivation

It is nothing new that the ability to predict, detect and diagnoses human diseases through medical image analysis is very promising to assist experts in clinical treatment. However, the task of viewing and interpreting these images is time-consuming and challenging. In the case of MRI brain, WML can occur at multiple sites, with varying shapes and sizes, and their image intensity profiles largely overlap with non-affected, healthy parts of the brain or other WML [67]. Furthermore, the amount of data is far too much for manual interpretation and has been one of the biggest obstacles in the effective use of MRI [59]. Thus, the analysis eventually became an exhausting and repetitive process task, even more when is necessary to combine several MRI types to accurately perform the diagnosis. The workload can lead to distractions and results in a misunderstanding of the diagnosis, especially when a patient simultaneously presents (1) large number and different types of anomalies and (2) parts of the same anomaly are present in several slices.

In addition, another difficult task in MRI analysis is the segmentation of WML. Studies have investigated this inherent variability in the manual segmentation of WML by trained specialists and reported volume ranging in until 68% [65].

Knowing all these issues in WML manual detection and segmentation, it is indeed important to develop computational techniques that assist the healthcare professionals and reduce misinterpretation and its consequences.

In this work, it is expected that an automatic WML segmentation software could assist radiologists. Preliminary results presented by it should not only reduce the workload, but also improve final results by offering a second opinion to the specialist. In addition, it is also expected that radiologist detects WML faster once the machine will do part of the job.

1.2. Objectives

1.2.1. General Objective

The general objective of this work is to present new computational methods for automatic detection of WML in different MRI modalities, using for this purpose, image analysis, clustering and machine learning techniques.

1.2.2. Specific Objectives

To achieve the general objective, it is necessary to fulfill the following specific objectives:

- Develop new computational methods for automatic detection of WML using only FLAIR MRI modality;
- Develop new computational methods for automatic detection of WML using FLAIR and T1 MRI modalities together, acquired from the same patient;

1.3. Work Organization

This work is organized as follows:

The Chapter 2 introduces the main concepts needed to understand this work.

The Chapter 3 presents four methods for WML detection. The first two methods use only the MRI of modality FLAIR. Both uses pre-processing techniques, SLIC0 for candidates segmentation, and CNN for classification, but the first one uses common CNN and simple patches while the second one uses parallel CNN with sets of multiscale patches. The last two methods use MRI modalities FLAIR and T1 together. Both uses pre-processing techniques, SLIC0 for candidates segmentation, and CNN for classification, but the first one uses parallel CNN and pairs of patches of T1 and FLAIR images while the second one uses common CNN with patches generated from pixelwise comparison of T1 and FLAIR images.

In Chapter 4, it is shown and discussed the results achieved through the application of the methods on the database. It is also shown some study of cases and the results are compared with related works.

Finally, in Chapter 5, it is presented the final remarks about this work. There, this work is evaluated, showing its contribution as well as proposing others as future works to achieve better results.

2. Theoretical Foundation

This chapter discusses the concepts necessary to understand the methods on next chapters. It starts explaining the DICOM standard, which is standard of the used database. Then, it is explained Clustering algorithms such as SLIC0, used here for candidates segmentation in MRI. Following, it is explained Image Registration used for better comparison of T1 and FLAIR MRI. Finally, it is explained Artificial Neural Networks and especially Convolutional Neural Networks which is used in this thesis for classifying WML candidates.

2.1. DICOM Standard

The word DICOM is an abbreviation for Digital Imaging and Communications in Medicine.

The DICOM standard was created by the American College of Radiology (ACR) and the National Electrical Manufactures Association (NEMA) in a committee in 1983. It emerged with the objective of standardizing the format of the images and their information used between the different manufacturers of computed tomography, nuclear magnetic resonance, ultrasound and so on. With this standardization, it would be possible to exchange information between physicians and specialists who are in different locations, especially with the use of Picture Archiving and Communications Systems (PACS), which allow the creation of databases [115].

The DICOM standard internally is a set of standards for treatment, storage and transmission of medical information in an electronic format, structuring a protocol. In other words, the DICOM standard is a set of rules that allow medical images and associated information to be exchanged between diagnostic imaging equipment, computers, and hospitals. The standard establishes a common language between equipment of different brands, which are generally not compatible, and between imaging equipment and computers, whether in hospitals, clinics or laboratories.

The DICOM standard should specify the following information: the set of protocols to be followed by the applications and equipment, the syntax and semantics of protocol commands, and associated information among others.

A DICOM file consists of a header and a body. The header contains information useful in interpreting the contents of the body of the file. You can find the size of the image, the level of compression, the date of the exam, etc. the body carries the image itself, which can be stored either in raw or compressed format in JPEG format.

In this thesis, the database used to validate our results is in DICOM standard.

2.2. Clustering

Grouping, also known as agglomeration or clustering, is a process by which individuals of a given sampling are divided into clusters obeying one or more criteria that compute the similarity between these individuals. An effective grouping should create groups so that individuals within the same group are as similar as possible and individuals from different groups are as different as possible. In other words, a clustering seeks to maximize interclass distance (distance between groups) and minimize intraclass distance (distance between individuals of the same group).

The grouping can be found under different names in different contexts, such as unsupervised learning (in pattern recognition), numerical taxonomy (in biology, ecology), typology (in social sciences), and partitioning (in graph theory). [116]

2.2.1. Superpixels

The concept of superpixels can be defined as a grouping of pixels based on one or more meaningful criteria. The purpose of this grouping is to capture the redundancies present in the image, forming atomic regions that which pixels carry related information and considerably reduce the complexity of image processing tasks.

The literature shows a variety of superpixel algorithms and their different applications, such as object segmentation [126-128], depth estimation [129], segmentation [130], body model estimation [131], and object localization [126]. In addition, a different segmentation approach has been widely used, including in segmentation problems in brain MRI. In this approach, pixels are grouped into perceptually meaningful atomic regions, taking into consideration that neighboring entities are usually highly correlated. One of the techniques related to this approach is the Simple Linear Iterative Clustering (SLIC), which uses a weighted distance measure that combines color and spatial proximity, while simultaneously providing control over the size and compactness of the superpixels [83].

However, if the image is smooth in certain regions but highly textured in others, SLIC produces smooth regular-sized superpixels in the smooth regions and highly irregular superpixels in the textured regions [84]. Therefore, in this thesis was used an improved version of SLIC, called SLIC0 [83], which adaptively chooses the compactness factor for each superpixel individually and generates regular shaped superpixels in both textured and non-textured regions alike, with high computational efficiency.

For a better understanding of the SLICO algorithm, the SLIC algorithm theory, its precursor, will be explained first. Then, the additional concepts that improved this technique to create SLIC0 are introduced.

2.2.1.1. SLIC

SLIC is a superpixels algorithm based on the K-means [112] algorithm that has as main advantage the reduction of the number of distance calculations by limiting the search space to a region proportional to the superpixel size. Given a color image in the CIELAB [112] color space with *N* pixels and a number of desired superpixels k, the clustering procedure begins with an initialization of the cluster center $C_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$, where l, a and b are the channels of an image in CIELAB color space and *x* and *y* are pixel coordinates. These clusters are sampled on a regular grid and composed by S pixels defined by:

$$S = \sqrt{\frac{N}{K}} \tag{1}$$

To avoid centralizing a superpixel in an edge region or centering on a noise pixel, its centers are moved to lower gradient locations in a 3 × 3 neighborhood. Then, each pixel i is associated with the center of the nearest superpixel in a centered area with a 2S x 2S extension, by a distance D. The distance measurement used by the SLIC is a 5-dimensional Euclidean distance in a labxy space, which combines color (l, *a* and *b*) and proximity distances between pixels (*x* and *y*) in a standardized way in a single distance. This measure is expressed by the equation:

$$D = \sqrt{{d_c}^2 + (\frac{d_s}{N_s})^2 * m^2} \tag{2}$$

where D is the distance measure used by the algorithm, m is a user-chosen compactness factor that ponders the relative importance between color similarity and spatial similarity, S the size of the superpixel (Equation 1), d_c is the measure of color distance between a pair of pixels i, j, expressed by the equation:

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2} \tag{3}$$

and d_s is a measure of spatial distance expressed by equation:

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{4}$$

2.2.1.2. SLIC0

The authors of the SLIC algorithm have observed that in superpixels covering larger regions, the contribution of the space distance component outweighs the contribution of the color distance component. This produces compact superpixels that do not adhere well to image boundaries. For smaller superpixels, the inverse is true. In regions of irregular texture, the color distance component stands out. Thus, with SLIC, it is a challenge to test values for the parameter m (compactness parameter) that best fit the problem that will be applied.

In order to circumvent this detail, the SLIC0 algorithm was proposed, which dynamically normalizes both the distance components of each superpixel through the maximum values obtained from each of them, for each particular superpixel. Mathematically, this can be expressed as:

$$D = \sqrt{(\frac{d_c}{m_c})^2 + (\frac{d_s}{m_s})^2} \tag{5}$$

where m_c corresponds to the maximum color distance observed on the superpixel and d_s corresponds to the maximum spatial distance observed in the superpixel. This new strategy produces superpixels that present a more regularized shape, regardless of the type of image.

Figure 1 illustrates the application of SLIC0 algorithm and compare the results with SLIC.

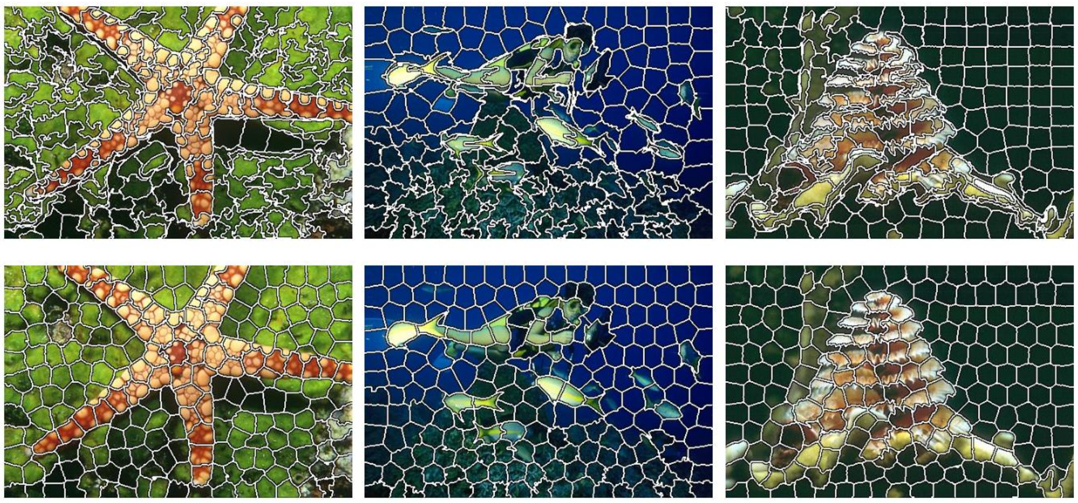


Figure 1: SLIC versus SLIC0. Top row of images shows SLIC output with a constant compactness factor for all superpixels, while the bottom row of images shows the output of SLICO, which chooses the compactness factor adaptively for each superpixel. [84]

The Equation 3 can be adapted to grayscale images. In this case, only light information is considered. This adaptation was used in this work, since the MRI does not have chromaticity components. This adaptation can be expressed through equation:

$$d_c = \sqrt{(l_j - l_i)^2} \tag{6}$$

As for the spatial distance component, this work was limited to using Euclidean distance in 2D, even the images being 3D. That is, the algorithm was applied slice to slice, separately. The justification for this choice is that the spacing between slices of all the MRI slices is very large, that is, in the z axis the information is not equally spaced with respect to the x and y axes.

2.3. Image registration

The literature uses several definitions for images registration. Brown [117] defines image registration as the process of transforming different sets of data into the same coordinate system. The data can be several photographs, data from different sensors, different times, or from different points of view.

Crum et al [118], in turn, explains the registration of images as a process for determining the correspondence between the characteristics of images collected at different times or using different imaging methods. Matches can be used to change appearance - by rotation, translation, stretching, etc. - from one image so that it more closely resembles the other so that the pair can be directly compared, combined or analyzed. The most intuitive use of registration is to correct different patient positions between scans.

Image registration consists of the following elements: two images, one fixed image and one moving image, a metric of similarity, a transformation function, an optimizer and an interpolator.

The metric of similarity is a numerical measure responsible for saying how much the moving image is similar to fixed one. The transformation function is the function that changes the moving image coordinates. This transformation function is calculated by the optimizer, optimizing some metric of similarity. Thus, the registration is an optimization problem, in which an optimizer estimates the transformation function that best maps the moving image to fixed image space, according to a value obtained by the metric of similarity.

Historically, registrations are divided into two groups: rigid registration and deformable (or non-rigid) registration. Rigid registration consider that the moving image is an object that needs to be rotated and / or translated so that there is a satisfactory match with the still image. That is, this type of register uses global transformation. The deformable registration, on the other hand, obtain this correspondence through some localized deformations. This in turn uses local transformation.

Registration is an essential process in several application areas linked to medical imaging. One common application is image-guided surgical procedure

[119], in which the registration of preoperative images and images obtained during surgery is of fundamental importance for the location of the pathology and the junction of different types of examinations, allowing you to group complementary information into one image. In addition, it has been used in segmentation, diagnosis and also in the temporal analysis of diseases, in which there is variation of patient positioning, requiring an alignment between images to optimize algorithms, among others.

2.3.1. Rigid registration

The applications of the rigid registration are simple ones which are summarized only by operations of rotation and translation, so that the moving image is not deformed, but placed in the same fixed image coordinate system.

2.3.1.1. Image Transformation Function

More formally, the transformation of coordinates of a moving image, M(x), into the coordinates of a fixed image, F(x), is given by Equation 7.

$$x' = T(x) = R_x(\Theta_x)R_x(\Theta_x)R_y(\Theta_y)R_z(\Theta_z)x + t \tag{7}$$

where R_x, R_y e R_z are the rotational matrices around the x, y and z axis respectively, considering an angle Θ and t is the translation vector. The rotation matrices are represented by Equations 8, 9 and 10.

$$R_x(\Theta_x) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\Theta_x & -\sin\Theta_x \\ 0 & \sin\Theta_x & \cos\Theta_x \end{pmatrix} \tag{8}$$

$$R_y(\Theta_y) = \begin{pmatrix} \cos\Theta_y & 0 & \sin\Theta_y \\ 0 & 1 & 0 \\ -\sin\Theta_y & 0 & \cos\Theta_y \end{pmatrix} \tag{9}$$

$$R_y(\Theta_y) = \begin{pmatrix} \cos\Theta_z & -\sin\Theta_z & 0 \\ \sin\Theta_z & \cos\Theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{10}$$

2.3.1.2. Image Interpolator

When the pixels of an image are transformed from one space of the image to another space, they may fall into positions that are not exactly a valid location in matrix of the image. Therefore, there might be some “empty spaces” in this matrix that must be estimated. This is estimation is achieved by using image interpolation of the transformed pixels.

The choice of the interpolation method can influence the smoothness of the metric of similarity space. Some of most common methods for interpolation in registration context are the linear, b-spIine, and nearest neighbor. In the case of this thesis, it is used linear interpolation.

Linear interpolation is commonly used because it is simple to understand and implement and is efficient in several problems. It is ideal to use it when there are few points to be estimated. In its operation, linear interpolation traces a line segment between two desired points and calculates its equation. Through this equation it is possible to estimate values between the extreme points of the segment, based on their location.

In relation to the images, linear interpolation traces a line segment between each pair of neighboring pixels (or voxels). Interpolation uses this segment to estimate values between neighboring pixels, which are the end points of the segment. This estimation is done based on the distance between the endpoints and their values. An interpolated pixel found exactly in the center of straight lines, for example, has a value equal to the arithmetic mean of the pixels of the endpoints.

Mathematically, linear interpolation can be defined as follows: being an interval [x_0, x_1] and the values f_0 and f_1 for the extreme points of the function $f(x)$, we obtain the interpolator polynomial defined in Equation 11.

$$f(x) = f_0 + \left[\frac{x-x_0}{x_1-x_0}\right]\ (f_1 - f_0) \tag{11}$$

2.3.1.3. Images Metric of Similarity

The metric is used by the optimizer to quantitatively evaluate the transformation parameters. Depending on the optimization method, some derivatives of the metric of similarity are determined as a function of the transformation parameters. The most common similarity metrics are quadratic mean, normalized correlation, and mutual information.

The quadratic mean is given by Equation 12. Given an N number of points in the image, the quadratic mean is the sum of the square of the difference between the intensity value of the movable image M, interpolated at point a: after transformation T, by value of intensity at point x in the fixed image F. As can be seen, the metric has some special characteristics, such as the simplicity of being calculated and the amplitude of the radius of variation. Its optimal value is reached when it becomes as close as possible to zero.

$$E(x|F,M,T) = \frac{1}{N}\sum_i^n \big(F(x_i) - M(x_i')\big)^2 \tag{12}$$

2.3.1.4. Registration Optimizer

The optimizer seeks for the transformation parameters that produce the best alignment between the fixed and moving images based on the similarity metric. The most used methods are descending gradient, the regular step descending gradient, the quasi-Newton, among others.

In this thesis it is used a variant of the descending gradient named steep descending gradient. It seeks to minimize an objective function. In this thesis this function is mean square error (EMQ).

The method starts from a starting point xo and then the gradient is calculated at that point. As the gradient is the direction of greatest variation for EMQ, the new point a is added with the negative of the gradient. In addition, a learning rate a is used to multiply the gradient obtained by limiting the size of the starting point offset. The result of the product is called a step or step and can be seen in Equation 13, which has the value of x in an iteration k.

$$x_{k+1} = x_k - \alpha \nabla EMQ(x_k) \tag{13}$$

In this work, the rigid registration was used to align the volumes of T1 and FLAIR images of the same patient. With this alignment, it is possible to combine the information from the exams at the voxel level to assist in detecting the WML in subsequent steps.

2.4. Artificial Neural Networks (ANNs)

The study of the brain in computation is interesting for enabling the development of biologically inspired information processing models, such as Artificial Neural Networks (ANNs). The RNAs represent an attempt to imitate the principles of brain functioning and apply them in computation models. ANN tries to simulate brain abilities such as adaption and self-organization, achieved through experience and/or learning, and high performance through parallel processing.

ANNs may also be defined as mathematical models that use a collection of simple computational units, called artificial neurons interconnected in a network. These models are commonly used in the pattern recognition task.

The works proposed in [124] were the first work involving ANNs and their peculiarities. According to Hafemann [120], the motivation for McCulloch and Pitts [124] to study neural networks was the fact that the human brain was superior to a computer in many tasks, an assertion that still remains today for tasks such as recognition of objects and faces, despite enormous advances in processing speed in modern computers.

2.4.1. Artificial neuron

Artificial neuron is the basic unit of an ANN and is used for building more powerful models. Explaining in a simple way, an artificial neuron is an element that receives signals and performs a weighted sum of them and that produces an output, according to an activation function. An artificial neuron can be expressed in mathematical terms by Equation 14.

$$f(x) = \sigma(\sum_{i=1}^{n} x_i w_i + b) \tag{14}$$

where xi is the input i, wi is the weight associated with the input i, b is the term bias and σ is the nonlinear function called activation function.

An artificial neuron implements this activation function as nonlinear function given its inputs, to provide an output. Some activation functions, which are commonly used in neuron models, are illustrated in Figure 2.

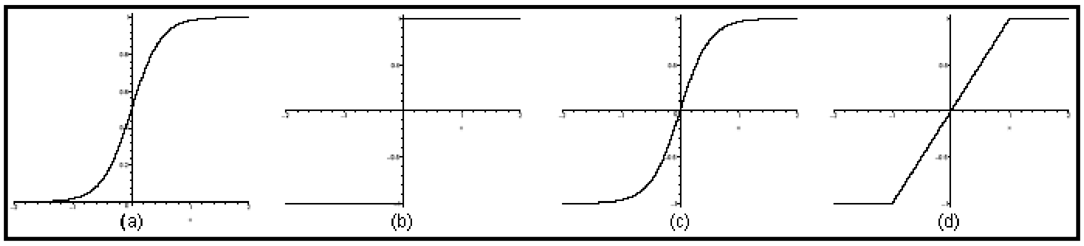


Figure 2: Activation functions. (a) sigmoid; (b) signal; (c) hyperbolic; (d) saturation.

Krizhevsky et al (2012) proposed a new and simple nonlinear function, called ReLU and, according to them, it converges up to 6 times faster than ANNs which use the hyperbolic tangent function. This function is presented in Equation 15.

$$\sigma(x) = max(0, x) \tag{15}$$

2.4.2. Multilayer Perceptron (MLP)

According to Hornik et al [125], the Multilayer Perceptrons (MLP) are universal approximators, that is, they can approach any measurable function to any desired degree of precision.

MLP is a very used ANN architecture. Basically, the MLPs are composed of an initial input layer, followed by one or more hidden / hidden layers, and the last layer would be the output of the network. Each layer is fully connected to its adjacent layer and produces an output vector according to the vector of the previous layer.

In order to "learn", the MLP passes in an inactive process of weight adjustments in the network connections. In this process, the network is first stimulated by the input information. An error is then calculated by some error function, comparing network output with the expected output. Following, this error is used to readjust the weights in the network using some learning algorithm. This process is repeated a number of times. So, the objective is to minimize this error by iteratively adjust network weights using the learning algorithm. The universal learning algorithm used by MLP is the backpropagation.

The output of each layer is calculated by applying the activation function of each neuron in all the neurons of the layer. Each neuron (except input ones) has a non-linear activation function. Equation 16 describes the output of a layer.

$$y_l = \sigma(W_l y_{l-1} + b_l) \tag{16}$$

where y_l is the output vector, W_l is the matrix of the weights assigned to each neuron pair of the layer l e $l-1$, e bl is the vector of the bias term of each neuron of the layer l.

2.4.3. Deep Learning

The depth of a neural network architecture refers to the number of nonlinear operations that make up this network. While many successful applications have used “shallow” networks (up to 3 layers), the organization of a mammal's brain is a deep-sea architecture [121].

Techniques based in Deep Learning can be visualized as a multi i-layered graph permeated with learning rules that build knowledge from hierarchical concepts. In other words, deep learning algorithms automatically extract complex data representations and develop a layered, hierarchical architecture of learning and representing data, where higher-level (more abstract) features are defined in terms of lower-level (less abstract) features. The literature points as the main advantage of its use the exclusion of step of designing and extracting a particular hand-crafted set of good features for a specific problem, task that can consume a large time portion with extensive experiments [86, 87]. Still, the three steps of feature extraction, selection and classification can be performed within the optimization of the same deep architecture which reduces the workload in classification problems [88].

According to Hafemann [120], deep neural networks have been investigated for decades, but deep training networks have consistently produced poor results until very recently. It has been observed in many experiments that deep nets are more difficult to train than “shallow” nets, and that deep nets often get trapped in apparent local minima when network parameters start randomly. However, Cireşan et al. [122] demonstrate that properly trained deep neural networks can achieve better results in many tasks.

According to Hafemann [120], the best published results for image classification use a type of architecture called the convolutional neural network.

2.4.4. Overfitting and Dropout

Deep neural network, with many parameters, are very powerful machine learning models. They can learn complicated relationships between their inputs and outputs. However, overfitting is a serious problem in such networks. When the amount of training data is limited, these complicated relationships can be generated because of sampling noises. These relationships, then, will exist in the training set, but not in the test set, which causes overfitting.

Many methods have been developed to reduce overfitting. One of the most famous are dropout. Dropout is a regularization applied to large neural networks, where each unit of a layer binds to all the units of the next layer. This simple technique was shown to reduce overfitting in neural networks [123]. One way to regularize a model and prevent overfitting would be to calculate the mean of the predictions of all possible parameter configurations, weighing each configuration by its posterior probability based on the training data. However, this is generally not feasible. With large neural networks, averaging the results of many separately trained networks is very expensive because it would require a very large amount of computation.

For a combination of models to be effective, the individual models must be different from one another. They must have different architectures and / or be trained with different datasets. Training many different architectures is difficult because finding optimal parameters for each architecture is a difficult task and training each large network requires a lot of computation. In addition, large networks typically require large amounts of training data. Because there will be multiple networks, even more data will be needed, and it may not be possible to divide that data into sufficiently large subsets for each of those networks. Yet even if it is possible to train many different large networks, using them all in the test is impractical in applications where a rapid response is important.

Dropout is a technique that addresses these issues. It avoids overfitting and provides a way to approximately combine many different neural network architectures efficiently. The idea is to temporarily remove hidden and visible drives in a neural network, along with all your inbound and outbound links. This process is represented in Figure 3. The choice of which drives will be removed is random. A simple way would be to keep each unit with a fixed probability p independent of other units.

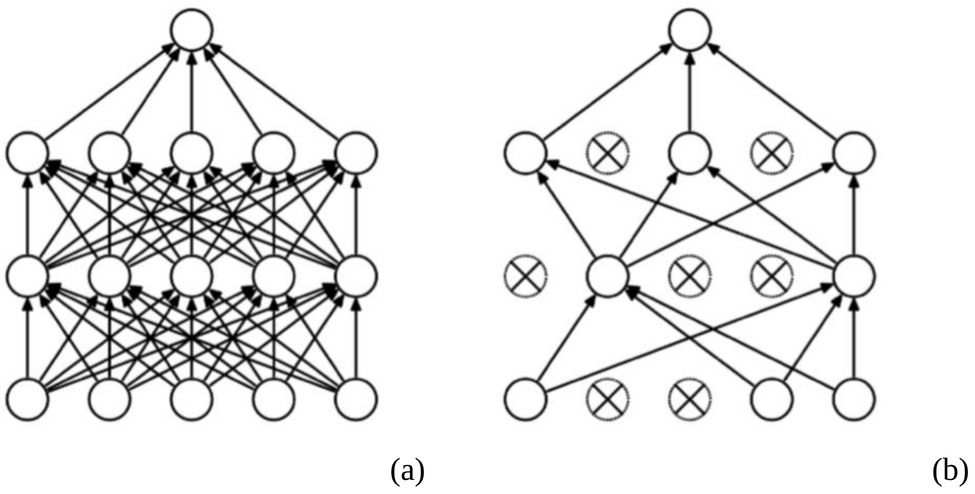


Figure 3: Illustration of the use of dropout in a neural network. (a) represents a common neural network with two intermediate layers and (b) represents a "net" network resulting from the dropout applied in (a). Source: [123].

2.4.5. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) is type of deep learning that achieved great results in image classification tasks. In general, the CNN consists mainly of convolutional layers, pooling (downsampling) layers, and fully-connected layers. The convolutional layers have trainable filters that are applied throughout the patch. Pooling layers are non-linear downsampling layers. The last layers of the network are fully-connected layers. The number of input neurons for the fully connected layers is defined by the number of pixels resulting from the layer prior to it. Since CNN are deep networks, it is also common to use dropout to prevent overfitted models, which are common in complex networks. An example of a CNN is shown in Figure 4.

Two major advantages can be highlighted from CNN: (1) large learning capacity, it is, make strong and mostly correct assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies) and (2) compared to standard feedforward neural networks with similarly-sized layers, it has much fewer connections and parameters and so they are easier to train [90].

There are four key ideas behind CNN that take advantage of the properties of natural signals: local connections, shared weights, pooling and the use of many layers. Units in a convolutional layer are organized in feature maps, within which each unit is connected to local input images in the feature maps of the previous layer through a set of weights shared called filter bank. The result of this local weighted sum is then passed through a non-linearity [91]. The pooling function replaces the output of a unit at a certain location with a summary statistic of the nearby outputs. Popular pooling function include the maximum or the average of a rectangular neighborhood, the L2 norm of a rectangular neighborhood, or a weighted average based on the distance from the central pixel [92]. Theses stages of convolution, non-linearity and pooling are stacked into two or more layers, followed by more fully-connected layer based in backpropagating gradients like multi-layer perceptron network. Thus, CNN consist of many convolutional layers, interspersed with pooling layers that reduce the dimensionality of the input signal, and usually a few fully connected layers and a final classification layer. The convolutional layers in such networks can be thought of as a feature extraction subsystem, not designed or selected by algorithm developers, but learned specifically for the task at hand during the training process [87].

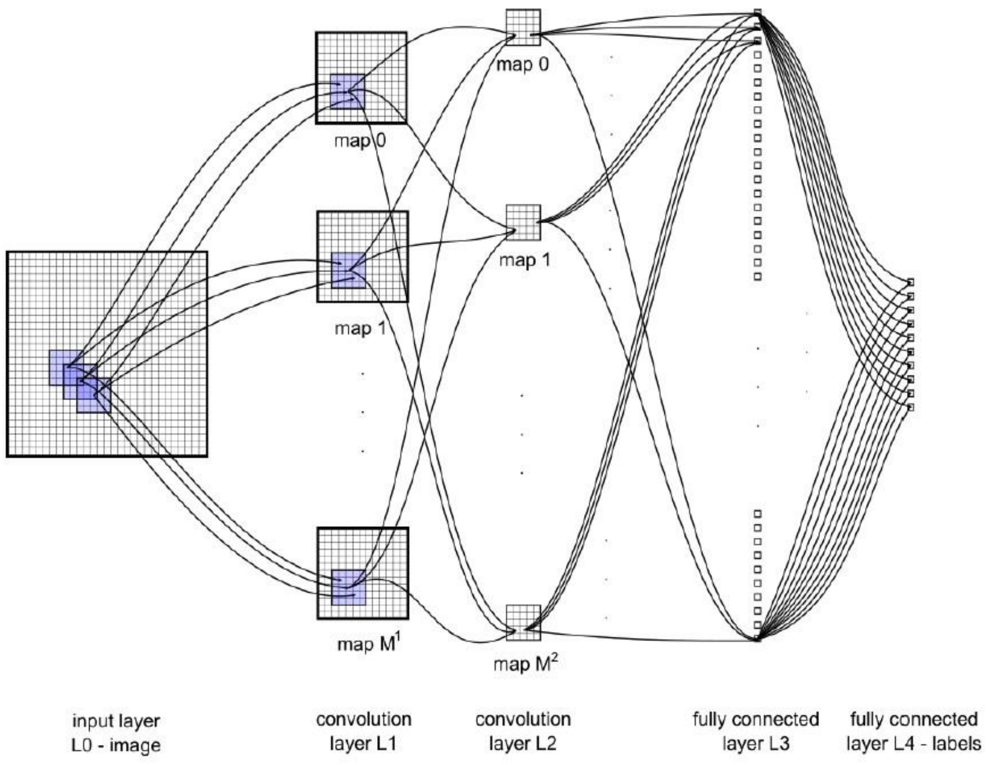


Figure 4: Example CNN Architecture. Source: [120].

The characteristics are extracted from one layer to another of the network, where a neuron in the current layer connects to a local receptive field in the previous layer. In this way, the characteristics are combined sequentially to extract higher-level characteristics.

2.4.5.1. Convolution Layer

The convolution layers are composed of trainable filters that are applied throughout the input image. For each filter, a neuron is attached to a subset of neuron from the anterior layer. For example, an image as input, the filters define a small area (3x3, 5x5, 7x7 pixels) and each neuron is connected only to neurons near the prior layer. The weights are shared between the neurons, causing the filters to learn the frequent patterns that occur in any part of the image. Figure 5 illustrates how convolution occurs in an image.

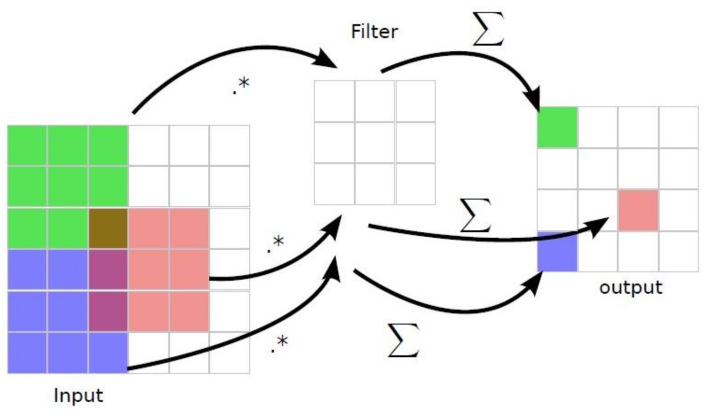


Figure 5: Convolution layer illustration. Source: [120].

Each workable filter is applied individually to each input image generating several feature maps, meaning each filter is responsible for detecting one type of feature in the image.

2.4.5.2. Downsampling layer

According to Hafemann [120], the downsampling (or subsampling or pooling) layers implement a non-linear solution reduction function to reduce dimensionality and capture small image invariance.

In the subsampling layer there is a reduction of the spatial resolution of the characteristic maps and in which the invariant characteristics of the displacements and distortions will be selected [113]. A much-used type of subsampling is shown in Figure 6.

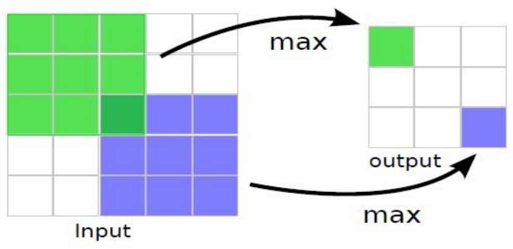


Figure 6: Illustration of the subsampling layer. Source: [120].

The type of subsampling shown in Figure 6 is known as maximum activation (maxpooling), where only the highest intensity pixel of the previous receptive field is maintained, that is, in a 2x2 mask with 4 values, only the highest pixel value among them will remain in the current image.

2.4.5.3. Fully connected layers

At the termination of convolution and subsampling layers sequences that have the task of extracting features from the image, the pixels of all feature maps of the previous layer are given as input to the fully connected layers, which in turn are responsible for the classification of patterns, like MLP.

In this thesis, CNN is used to classify the candidate regions as WML and non-WML.

2.5. Validation Metrics

Validation metrics are metrics that are used to quantitatively evaluate the effectiveness of a classification model. Through these metrics it is possible to compare models generated from different algorithms or from same algorithms with different parametrization. Therefore, they are essential in searching the best models when developing a method or tuning its parameters. Also, they are used to compare and rank results from different works, so they are the judges that establish the state of art.

For calculating the validation metrics, the classification model is first used for predicting the classes of a set of samples. This set of samples is named test set and their actual classes are known a priori. The classes predicted by this model for each sample are then compared with their actual classes. So, the validation metrics are calculated using this comparison of predicted classes and actual classes.

To validate the methods proposed in this thesis it is used the following metrics: sensitivity, specificity, accuracy. They are described respectively in Equations 17, 18 and 19.

$$sen = \frac{TP}{TP+FN} \tag{17}$$

$$spe = \frac{TN}{TN+FP} \tag{18}$$

$$acc = \frac{TP+TN}{TP+TN+FP+FN} \tag{19}$$

where:

- *TP* is the number of True Positives (positive samples classified as positive samples);

- *TN* is the number of True Negatives (negative samples classified as negative samples);
- *FP* is the number of False Positives (negative samples correctly as positive samples);
- *FN* is the number of False Negatives (positive samples classified as negative samples);

Sensitivity measures the ability of the model to accurately predict the presence of one condition. In our case, it measures the percentage of WML regions that are correctly classified as WML regions.

Specificity measures the ability of the model to accurately predict the absence of the condition. In our case, it measures the percentage of non-WML regions that are correctly classified as non-WML regions

Accuracy is the weighted mean of sensitivity and specificity. So, it measures the ability of the model to accurately predict both presence and absence of one condition. In our case, it measures the percentage of regions that are correctly classified as their actual classes, be it WML region or non-WML region.

3. Methods

In this chapter, it is shown the four proposed methods. The first two methods are applied in the context where there is only one volume of modality FLAIR available for each patient. The last two methods are applied in the context where there are two volumes, one of modality FLAIR and other of modality T1, for each patient. These methods are explained in more details in next sections.

3.1. Method 1

The first method for WML detection uses only FLAIR MRI. It is divided into four steps as described in Figure 7. In brief, the first step, image acquisition, details the used materials. In the second step, image preprocessing, the preprocessing is performed for skull stripping and image enhancement. In the third step, candidates segmentation, the SLIC0 algorithm is applied in the image for segmenting WML and non-WML regions. Lastly, the fourth step, candidates classification, the convolutional neural network is trained using the patches generated from segmented regions and, at the end, those regions are classified. The next subsections explain in more detail these steps.

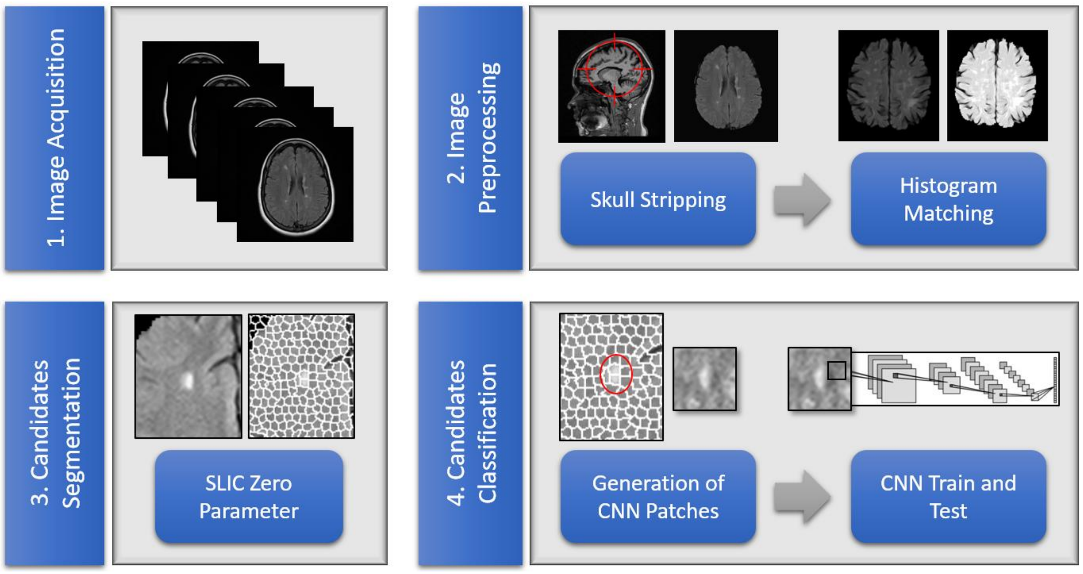


Figure 7: Method using only FLAIR MRI.

3.1.1. Image Acquisition

The MRI database used in this work is private and was provided by a diagnosed medical company called DASA [141]. It is composed of 93 cases of FLAIR MRI, obtained from different MRI devices using different parameters. All images were extracted in the axial plane. Each slice has dimensions around

500×500, with each volume having between 20 and 30 slices. The slice spacing varies between 4 and 5 mm.

In all database cases, WML of different natures are found in the white matter of the brain. In total, 4499 WML distributed among the 93 cases are found, with some containing less than 10 WML while others containing more than 100 WML. Therefore, this database can be considered quite heterogeneous, which helps to validate the generality of the proposed method.

The images were gathered with support of radiologists. From 93 images, 40 were marked with help of the specialist, using a software developed specially for this purpose (Appendix A). The other images, however, were used just for training, so the test results are reliable. The specialist was instructed to avoid marking WML with diameter smaller than 1mm since they are medically irrelevant.

3.1.2. Image Preprocessing

This section seeks to improve the images so that the next steps can be performed more efficiently. Following, the subsections are explained.

3.1.2.1. Skull Stripping

Head MRI includes not only brain tissue, but also other tissues present in patient's head and neck. So, one good approach is to remove them a priori to emphasize the brain. The literature presents several algorithms for this purpose [104–110]. Probably the most used tool is the Brain Extraction Tool (BET), developed by Smith et. al. [110]. However, in this work, we used the brain segmentation algorithm of Bauer et. al. [111], because it is simpler, less time-consuming and particularly effective in brain MRI that presents WML. The result of this segmentation is illustrated in Figure 8.

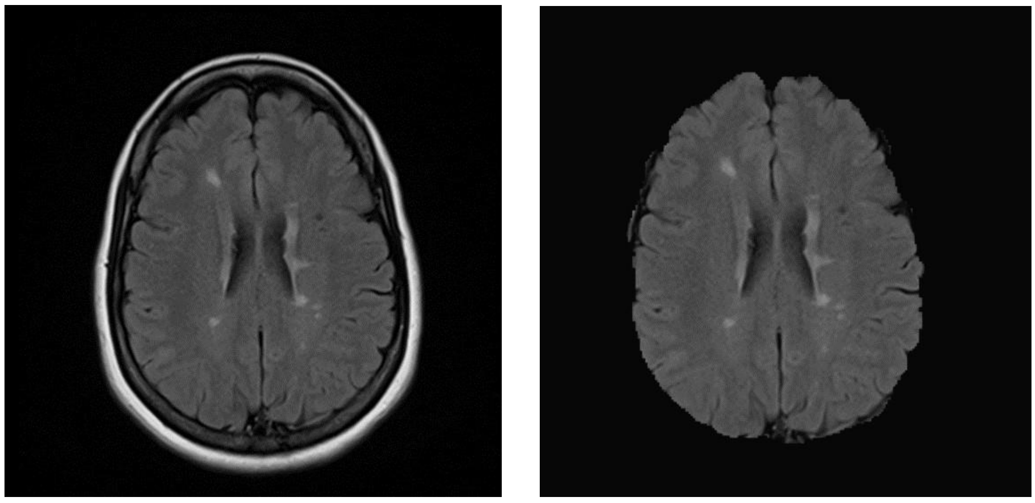

(a) (b)

Figure 8: Brain segmentation in FLAIR MRI. (a) FLAIR MRI before segmentation. (b) FLAIR MRI after segmentation.

3.1.2.2. Histogram Matching

Since the images were obtained from different devices and with different acquisition configurations, similar regions of the brain may have very different intensities from patient to patient. This distinction can disrupt an eventual comparison of texture features of brain regions. To alleviate this kind of issue, Histogram Matching [112] algorithm is commonly applied, using a single MRI volume of the database as a reference for all others. Figure 9 illustrates its application.

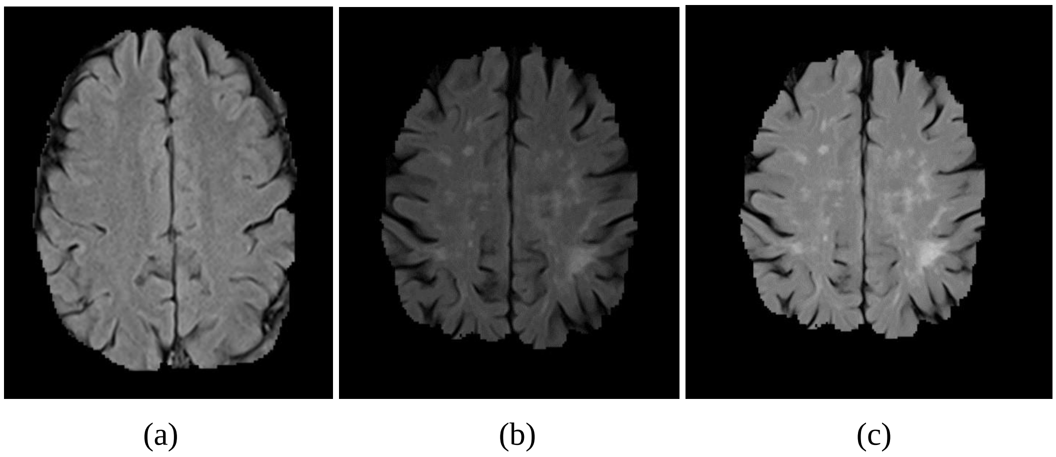


Figure 9: Histogram Matching. (a) FLAIR MRI used as reference. (b) FLAIR MRI with histogram to be adjusted. (c) FLAIR MRI with histogram adjusted based in (a).

3.1.3. Candidates Segmentation

At this step, the WML candidates will be segmented. For this, it is used the SLIC0 algorithm (Subsection 2.2.1.2). This algorithm groups nearby sets of neighboring pixels based on spatial distance and intensity. A group of pixels is called a superpixel. Figure 10 shows the result of the SLIC0 applied to a slice of the brain. Superpixels formed on regions outside brain are disregarded.

The algorithm used in this case is 2D, applied slice by slice. It was preferred to apply it this way because the spacing between slices is much greater than in the other axis. This would degrade the formation of supervoxels if applied in 3D, so that 2D results are much better. In addition, the slice spacing is also large enough that most of the WML were present in only one slice. The only parameter of this algorithm is the desired number of superpixels.

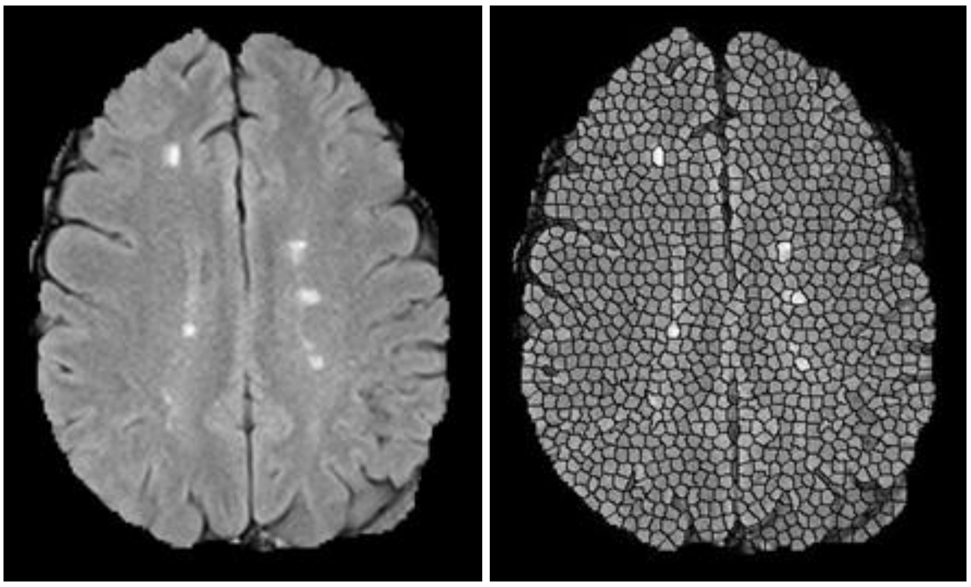


Figure 10: SLIC0 applied in FLAIR MRI. (a) Original FLAIR MRI Slice. (b) FLAIR MRI slice after SLIC0 execution.

It was empirically observed that, for the dimensions and spacing of the database images, 3500 superpixels per slice (including those located at the background of the image) would be a reasonable amount. Doing so, the superpixels have their dimensions like a microangiopathy or other type of small WML. This also minimizes the chances of a superpixel originally assigned to a small WML being too large that it could add pixels beyond the WML. In contrast, larger WML could be composed by more than one superpixel.

3.1.4. Candidates Classification

After segmentation of the candidates, the next step is to classify them. The candidates (superpixels) are classified in WML regions or non-WML regions. For this, it is used the machine learning technique CNN (Subsection 2.4.5). Following it is explained in more details how the CNN input data were created and how the training and testing procedures were designed and executed.

3.1.4.1. Generation of Patches

Many machine learning techniques use attributes extracted from the images as input. In such cases, for good results, good attribute engineering is required so that they are representative enough to discriminate the image class. One of CNN’s advantages is that it does not need an explicit attribute extraction step. Instead, the neural network uses the image itself as input and attempts to implicitly extract the best attributes.

The candidates in this method are superpixels. Therefore, it is necessary to generate images (called patches, in CNN context) that represent them and that are suitable as inputs for CNN. A possible patch to represent a superpixel would be composed by pixels inside its bounding box. However, due to the resolution of the MRI, and the dimensions adopted for a superpixel, such patch would be too small, with few pixels. This would be particularly risky in a CNN architecture, where convolutions and downsamplings are applied and, consequently, there is a decrease in number of pixels.

Also, a good patch should carry enough significative information to represent the candidate. Using larger input images would capture the neighborhood behavior of the WML. This is interesting because it is known that the tissue around the WML tends to have more hypointense values than the WML tissue itself, so capture such a characteristic can improve CNN performance.

So, knowing that a patch should have a satisfactory number of pixels and should carry meaningful information to represent WML and non-WML regions, it was decided that a patch should be generated through the bounding box of the region composed by a superpixel and its neighbors superpixels.

One of the requirements of CNN is that the patches have the same dimensions. However, this is not always done effectively, especially in cases where the patches are generated from segmentation algorithms, whose segmented regions may assume different dimensions. In the case of this proposed method, the use of SLIC0 superpixels help to overcome the problem of dimensions. Superpixels have similar dimensions and, consequently, their bounding boxes have similar dimensions. The small difference in dimensions is corrected by searching the largest bounding box from all superpixels neighborhoods in database. Then, the largest dimensions are used to generate all patches with the same size.

The procedure to generate CNN patches is illustrated by Figure 11.

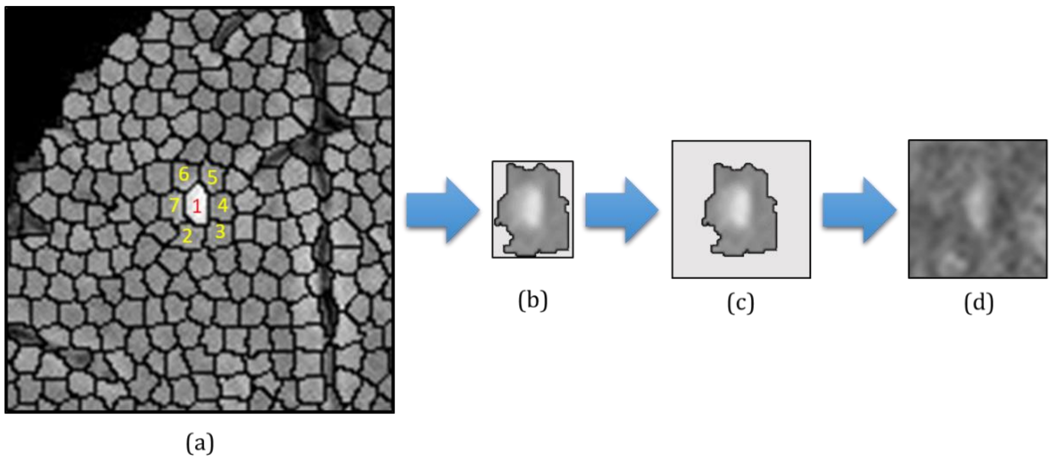


Figure 11: Generation of CNN patches. (a) represents the superpixels of a slice of the brain. The superpixel marked by the number 1 is the one you want to generate

an patch on CNN. The superpixels marked by numbers 2 to 7 are their neighbors. (b) represents the bounding box of the superpixels marked by numbers 1 to 7 in (a). (c) represents the bounding box corrected using the largest dimensions of all bounding boxes from superpixels neighborhoods in database. (d) represents the patch generated from the pixels inside of the corrected bounding box.

3.1.4.2. CNN Train and Test

For the training and testing, the patches need to be divided into two groups: those who represent WML regions and those who represent non-WML regions. To define this, it is used the WML markings. A superpixel is a WML region if it has at least 10% of its pixels found in any WML marking. The Figure 12 illustrates how this procedure is done.

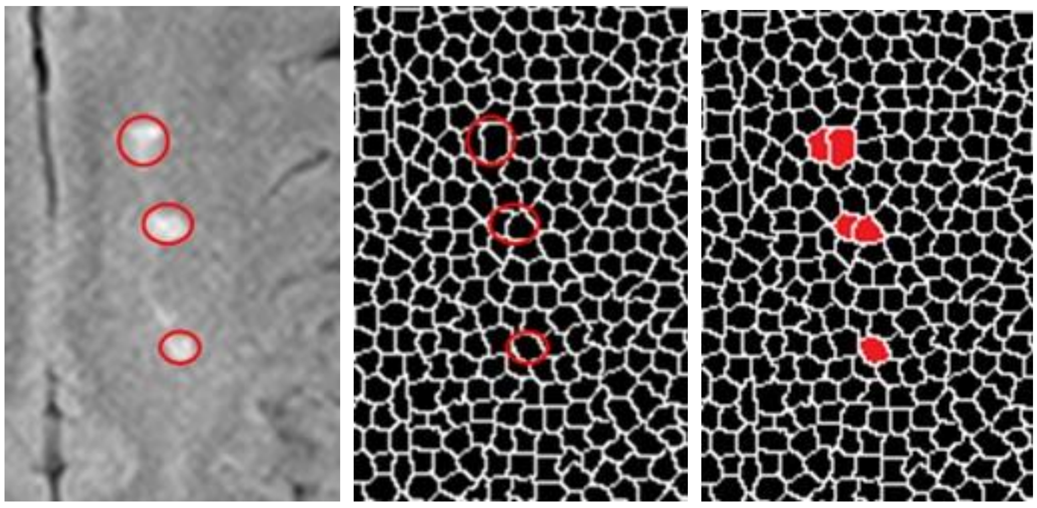


Figure 12: Selecting WML superpixels. The image (a) represents a slice with three WML, circled by their markings (red color). The image (b) represents the superpixels generated by applying the SLIC0 in (a). In image (c), superpixels having at least 20% of their pixels present in any of the markings of a WML of (b) are painted red.

Figure 12 shows that there are superpixels that have pixels presented in some marking but are not considered a WML region. This happens because, empirically, it was observed that superpixels having less than 10% of their pixels present in some marking has more characteristics of non-WML region. Once you know which superpixels are WML regions, all other superpixels are considered non-WML regions.

After dividing patches of WML and non-WML regions, it is now necessary to separate the training and validation databases for the training step and training and test databases for the test step. For each of these databases, patients MRI were selected randomly and patches were stored. However, for test and validation datasets, the MRI were randomly selected from those marked by the specialist.

Finally, after defining the datasets, it is necessary to define the CNN architecture to submit the patches. The best configurations of the network were selected after many training sessions. In each session, a new configuration of the network is experimented. The generated model of each session is used to classify the validation database. Finally, the configuration that generated the model that achieved the best results is selected. Then, the selected configuration is used to train the training database in the test step. The result of this training produces the final model. This model will be used to classify superpixels as WML regions or non-WML regions through their respective patches from the test database.

All training sessions, however, shared some parameters. It was used 100 epochs as a stopping criterion, learning rate value 0.001, momentum value 0.9 and input batch of 64 images. As for the patch dimensions, they depend on the dimensions of the bounding of superpixels neighborhoods, as explained in Subsection 3.1.4.1. For convenience, the calculated dimensions of the patches are 45×45 pixels.

The network that resulted in the best model consists of the following sequence of layers: (1) three convolutional layers, (2) one pooling layer, (3) other three convolutional layer, (4) other pooling layer and the fully-connected layers (MLP) composed of: (5) one input layer, resulting from the output of (4) flattened in a 1D vector, (6) one hidden layer and (7) an output layer.

Knowing the sequence of layers of this network, their internal configuration are as follow: (1) use 20 filters of size 3 × 3 and ReLU activation function; (2) uses maxpooling function of size 2×2 and ReLU activation function, (3) use 50 filters of size 3×3 and ReLU activation function; (4) uses maxpooling function of size 2×2 and ReLU activation function; (5) connects with (6) using a dropout function with 0.5 chance of activation; (6) contains 500 neurons and ReLU activation function and (7) contains two neurons and uses a softmax function giving the probability of the input to belong to either class.

This architecture is illustrated in Figure 13.

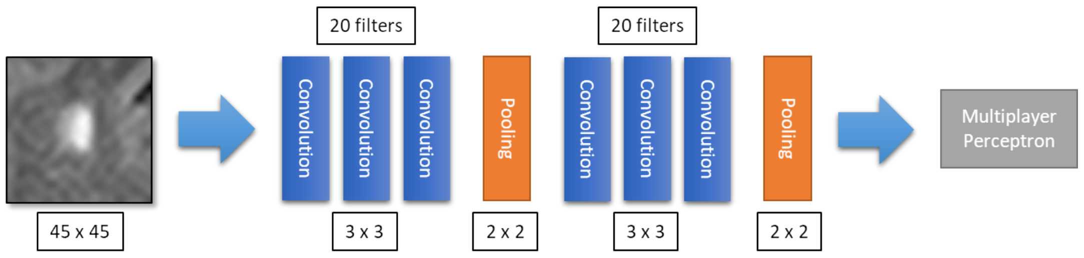


Figure 13: Architecture of the CNN of the first method.

3.2. Method 2

The second proposed method for WML detection uses only FLAIR MRI such as the Method 1 (Section 3.1). It is divided into four steps as described in Figure 14. The first three steps are the same from the Method 1. In fourth step, however, it is proposed the use of a parallel CNN which uses multiscale patches as input data. The next subsections explain in more detail the fourth step of this method.

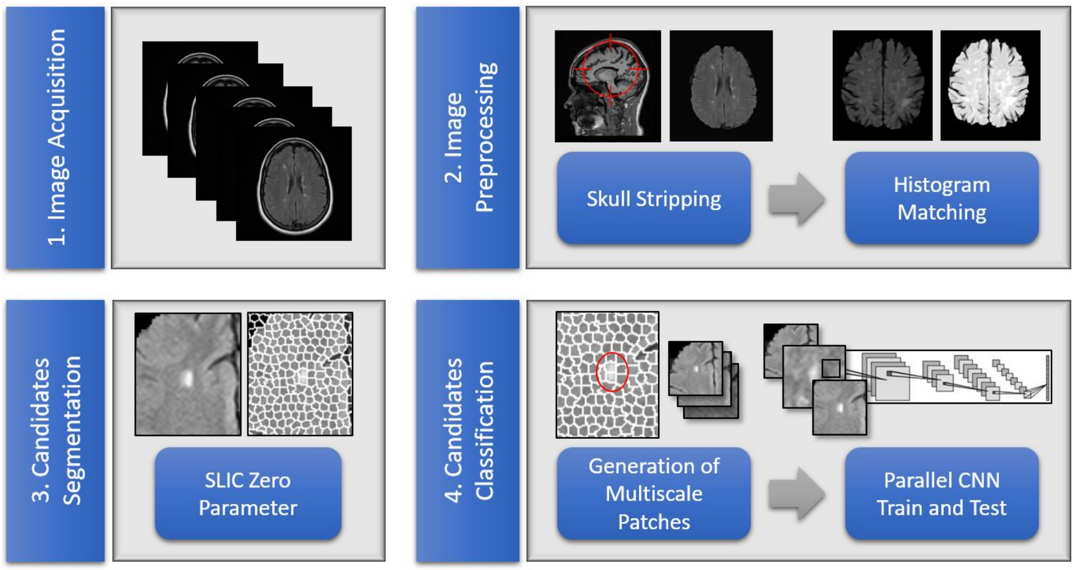

1. Image Acquisition
2. Image Preprocessing
Skull Stripping
Histogram Matching
3. Candidates Segmentation
SLIC Zero Parameter
4. Candidates Classification
Generation of Multiscale Patches
Parallel CNN Train and Test

Figure 14: Method using only FLAIR MRI.

3.2.1. Candidates Classification

After segmentation of the candidates, the next step is to classify them. Following, it is explained in more detail how the CNN entries were created and how the training and testing procedures were designed and executed. It is also explained how parallel CNN with multiscale patches approach works and its advantages.

3.2.1.1. Generation of Multiscale Patches

After superpixels segmentation, it is necessary to generate images that represent them in CNN. For this generation, we first calculate the centroid for each candidate superpixel. This centroid is calculated using the Equations 20 and 21.

$$x_c = \sum_{i=1}^{n} x_i / n \tag{20}$$

$$y_c = \sum_{i=1}^{n} y_i / n \tag{21}$$

where:

- x_c is the *x* coordinate of the centroid of the superpixel;
- y_c is the *y* coordinate of the centroid of the superpixel;
- x_i is the *x* coordinate of the *ith* pixel of the superpixel;
- y_i is the *y* coordinate of the *ith* pixel of the superpixel;
- *n* is the number of pixels.

Once the centroid of each candidate superpixel is calculated, it is generated three patches with different sizes, but all centered in superpixel centroid. These patches have sizes 42 x 42 pixels, 84 x 84 pixels and 126 x 126 pixels, respectively, so the second patch has two times the size of the first patch and the third patch have three times the first patch.

The choice of patches sizes was done empirically. However, we tried to create patches that are large enough for our CNN architecture but small enough to avoid meaningless background pixels or pixels outside image. After multiscale patches were created, the largest ones are resized so that they have the same size of the smallest (42 x 42 pixels). The procedure to generate multiscale patches is illustrated by Figure 15.

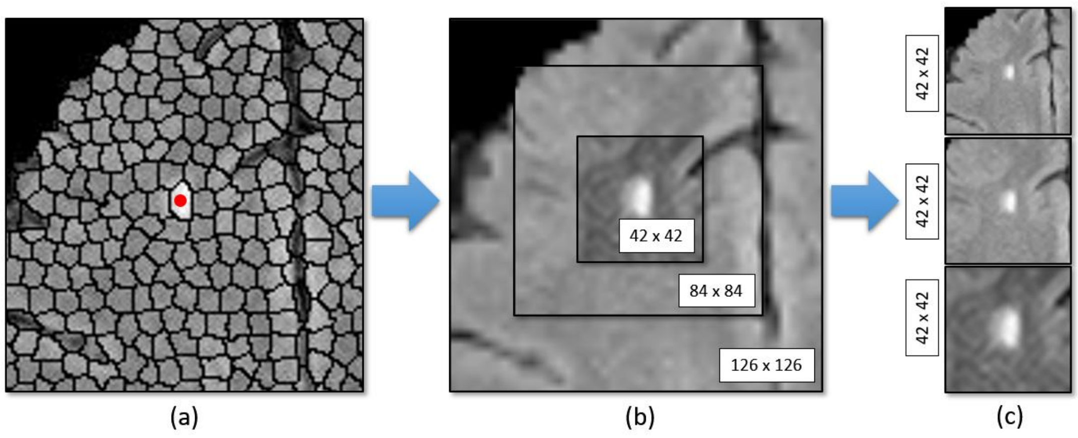


Figure 15: Generation of multiscale patches. The red dot in (a) represents the centroid of a superpixel. In (b), it is shown the three multiscale patches centered in this centroid. In (c), the largest patches are resized, so they have the same size of smallest

3.2.1.2. Parallel CNN Train and Test

After generating the patches for each superpixel, it is now necessary to separate patches as patches WML and non-WML regions and define the databases for training and test steps. This is all done such as Subsection 3.1.4.2, but this time there are three multiscale patches for each superpixel, as explained in Subsection 3.2.1.1.

The network used in this method is a parallel CNN. A Parallel CNN can be defined as a CNN that contain two or more independent paths of convolution and downsampling layers. Each path should receive a different patch as input. However, the last layers of these paths are merged in a single input layer for a MLP. The main idea of a Parallel CNN is to extract features of different patches that represents the same object independently but combine these features as an input for a more robust training.

One of the limitations of patch-based CNNs is that patches do not carry spatial information. So, the CNN relies only of the local pixels that are present in the patch and do not see around it. In other words, CNN does not have a peripheral vision like human eyes.

When the human eye is focusing on a certain point in an image, this point is clearer and carries more meaningful information. However, the points around are also captured and also provides useful information of the whole context, although they are less relevant and more unclear than the point we are focusing. It is also perceptive that the more a point is distant from the point the eye is focusing, the more unclear this point is.

The idea behind a Parallel CNN that uses multiscale patches is to work more like the human vision. So, like a human eye, it seeks to aggregate more spatial awareness of the candidate while focusing more on pixels that are nearer the center of the candidate. As explained, the smallest multiscale patch is not reduced. So, it contains all pixels from the image, not losing any information. On the other hand, the second larger patch becomes more unclear or “blurred” since it is reduced, losing information. This also happens to the largest patch, but it loses even more information due to the greater reduction. So, it is expected that parallel CNN with multiscale patches works better than a common CNN. However, it is also expected it is much slower than common CNN due to the increased number of parameters.

Training sessions were executed to find the best parallel CNN parameters. All training sessions, however, shared some parameters. It was used 100 epochs as a stopping criterion, learning rate value 0.001, momentum value 0.9 and input batch of 64 images.

The architecture that resulted in the best model is as follows. Each path of convolution and downsampling layers consists of the following sequence of layers: (1) three convolutional layers, (2) one pooling layer, (3) other three convolutional layer, (4) other pooling layer. The outputs layers of these paths are merged and flattened to be the input layer of the MLP. MLP is composed of the following layers: (5) one input layer, resulting from the output of (4) merged and flattened in a 1D vector, (6) one hidden layer and (7) an output layer.

Knowing the sequence of layers, their internal configuration are as follow: (1) use 20 filters of size 3×3 and ReLU activation function; (2) uses maxpooling function of size 2×2 and ReLU activation function, (3) use 50 filters of size 3×3 and ReLU activation function; (4) uses maxpooling function of size 2×2 and ReLU activation function; (5) connects with (6) using a dropout function with 0.5 chance of activation; (6) contains 500 neurons and ReLU activation function and (7) contains two neurons and uses a softmax function giving the probability of the input to belong to either class. This architecture is illustrated in Figure 16.

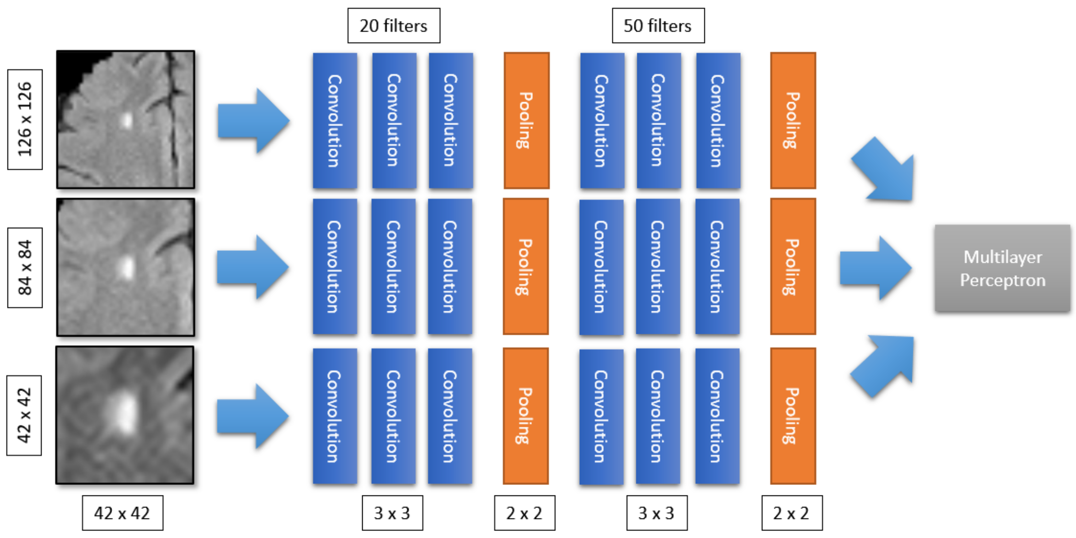


Figure 16: Architecture of the parallel CNN with multiscale patches.

3.3. Method 3

The third proposed method for WML detection uses pairs of FLAIR and T1 MRI. It is divided into four steps as described in Figure 17. Some steps are the same as the Method 1 (Section 3.1), but this time they consider both FLAIR and T1 MRI.

In brief, the first step, image acquisition, details the materials used. In the second step, image preprocessing, the preprocessing is performed for better use and compare of the T1 and FLAIR images. In the third step, candidates segmentation, the SLIC0 algorithm is applied to the FLAIR images for segmenting WML and non-WML regions. Lastly, the fourth step, candidates classification, the CNN is trained using the pairs of patches generated from superpixels of T1 and FLAIR MRI combined. The next sections explain in more detail these steps.

3.3.1. Image Acquisition

The MRI database is private and was provided by DASA such as the first method database (Section 3.3.1). Actually, this database is a subset of the first method database, composed only by the patients who contains both FLAIR and T1 MRI. So, it is composed of 73 cases of pairs of FLAIR and T1 MRI. The FLAIR images were acquired in axial plane while T1 images were acquired in sagittal plane. The images of this database were obtained from different magnetic resonance imaging devices using different parameters. Each slice has dimensions around 500×500, with each volume having between 20 and 30 slices. The slice spacing varies between 4 and 5 mm.

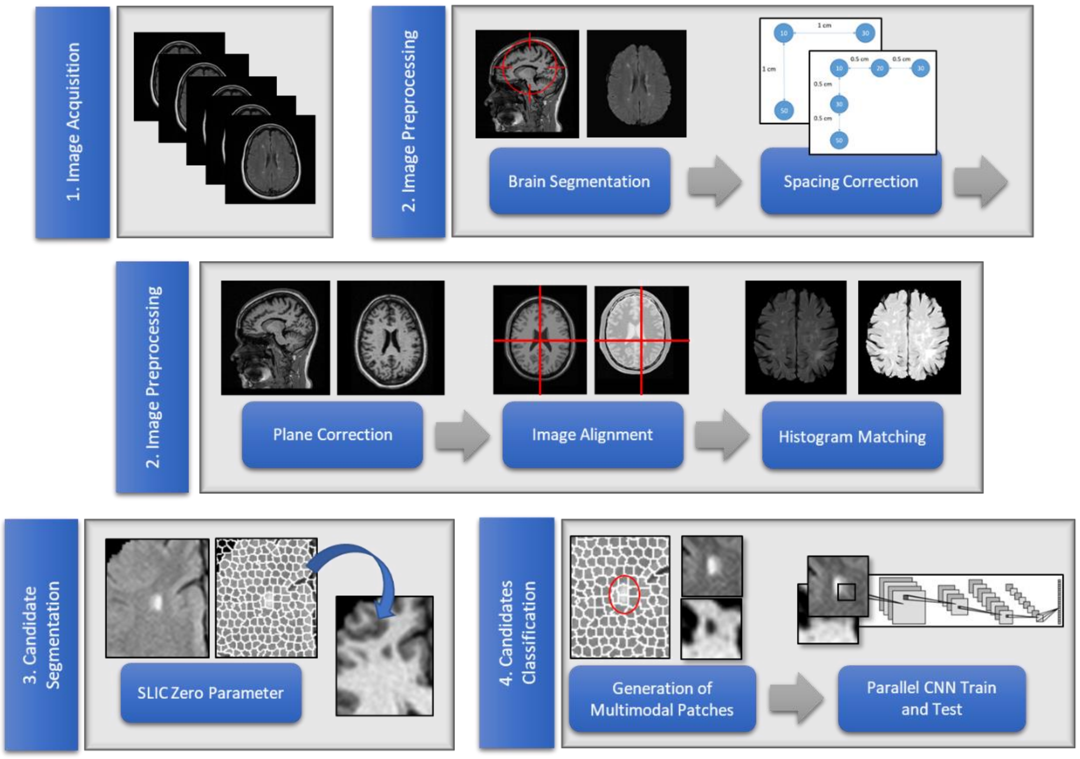


Figure 17: Method using T1 and FLAIR MRI.

In all database cases, WML of different natures are found in the white matter of the brain. In total, 3831 WML distributed among the 73 cases are found, with some containing less than 10 WML while others containing more than 100 WML. Therefore, this database can be considered quite heterogeneous, which helps to validate the generality of the proposed method.

The images were gathered with support of radiologists. From 73 images, 25 were tagged with help of the specialist, using a software developed specially for this purpose (Appendix A). The other images, however, were used just for training, so the results can be considered reliable. The markings are the same from the first database (Section 3.3.1).

3.3.2. Image Preprocessing

This section seeks to improve the images so that the next steps can be performed more efficiently. Following, the subsections are explained.

3.3.2.1. Skull Stripping

As explained in Subsection 3.1.2.1, it is necessary to segment the brain, eliminating other elements of the MRI. Here, it is also used the Bauer et al algorithm [111], but it is applied in both T1 and FLAIR MRI. Figure 8 (Subsection 3.1.2.1) shows the result of the brain segmentation in a FLAIR MRI and Figure 18 shows the result of the brain segmentation in a T1 MRI.

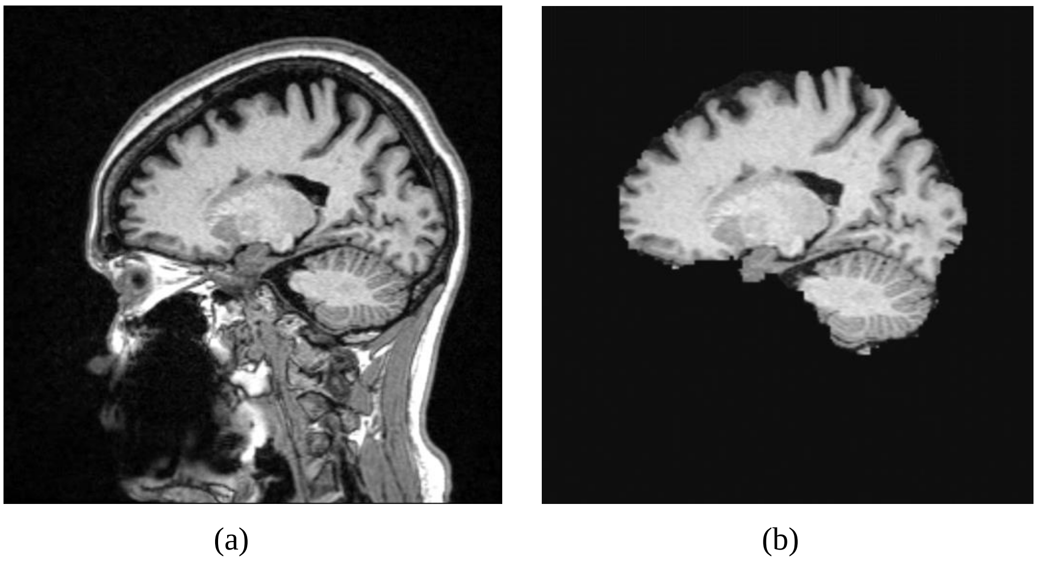


Figure 18: Brain segmentation. (a) T1 MRI before segmentation. (b) T1 MRI after segmentation.

3.3.2.2. Spacing and Plane Correction

Since T1 and FLAIR MRI were acquired with different protocols, there are differences in their voxels spacing and planes. As explained in Subsection 3.3.1, FLAIR MRI are in axial plane and T1 MRI are in sagittal plane, but for voxel-wise analysis between then, it is desired they are in same plane and have the same spacing. Knowing that, it was decided to resample T1 images to axial plane for better comparison. The voxels spacing of T1 images were also corrected to match with the FLAIR images using linear interpolation (Subsection 2.3.1.2). The result of this transformation is showed in Figure 19.

3.3.2.3. Image Alignment

Next, to reduce the area of study of the images and thus reduce computational time of the next steps, they were cropped to their segmented brain 3D bounding box as shown in Figure 20. This operation was applied in both FLAIR and T1 images.

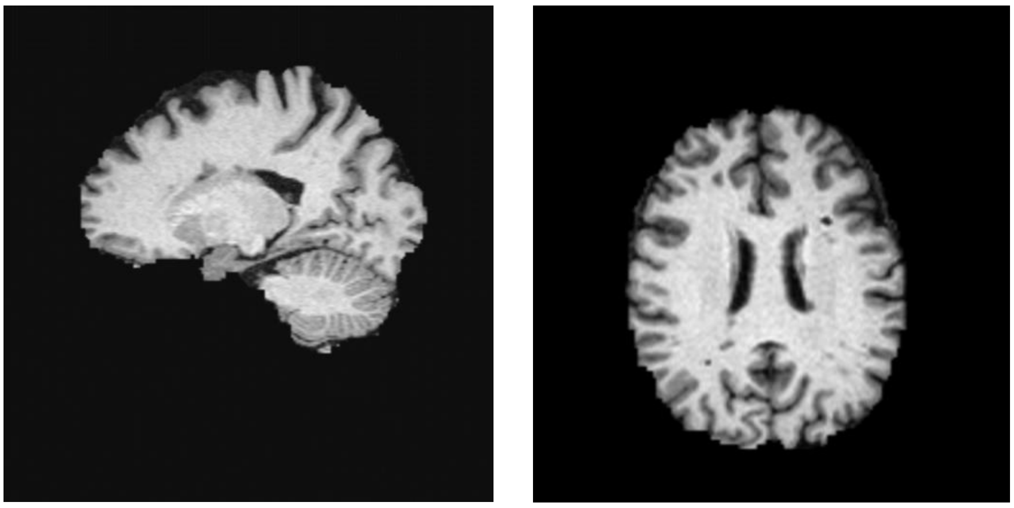


Figure 19: Resampling T1 MRI to axial plane. (a) T1 MRI before resampling. (b) T1 MRI after resampling.

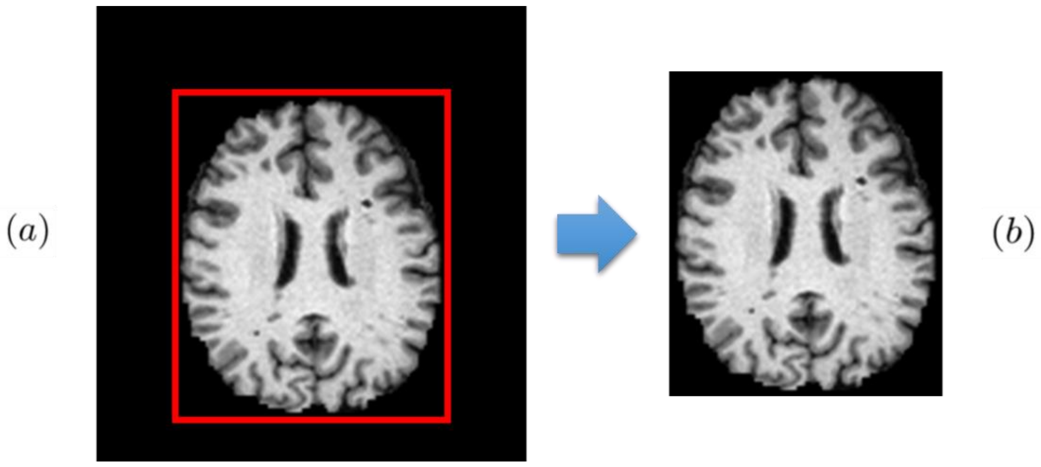


Figure 20: Cropping images to their segment brain 3D bounding box. (a) T1 slice before cropping. The red lines in (a) represents the 3D bounding box of the brain on that slice. (b) T1 slice after cropping.

Once images are composed only by brain voxels and are also in same plane, spacing and dimensions, it is now necessary that their voxels are matched in the same coordinates for voxel-wise comparison. For that, it is used 3D rigid registration (Subsection 2.3.1). T1 images are used as fixed images while FLAIR images are used as moving images. The Figure 21 illustrates a registration.

3.3.2.4. Histogram Matching

The next step is to normalize the intensity of the images. As explained in Subsection 3.1.2.2, to alleviate the inhomogeneity among MRI histograms in a database, the histogram matching algorithm is commonly used. However, in this subsection, it is applied in both T1 and FLAIR MRI.

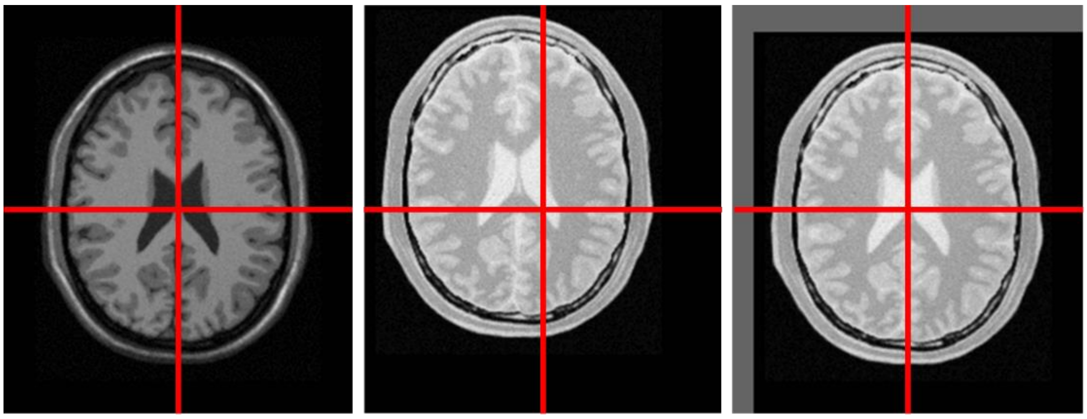


Figure 21: Aligning T2 and T1 images using 3D rigid registration. (a) T1 slice. (b) T2 slice before registration. (c) T2 slice registered. The red lines in images helps to see the alignment before and after registration.

3.3.3. Candidates Segmentation

In this section, the WML candidates will be segmented using SLIC0, such as in Section 3.1.3. In this case, SLIC0 is only applied in FLAIR images since they are better to highlight tissue anomalies than T1 images. The resulting superpixels from FLAIR images are then used to label voxels in both T1 and FLAIR images. Since T1 and FLAIR images are aligned by rigid registration, it is expected that superpixels correspond to the same tissue areas in both images.

3.3.4. Candidates Classification

After segmentation of the candidates, the next step is to classify them as WML regions or non-WML regions using CNN. Following, it is explained in more details how the CNN entries were created and how the training and testing procedures were designed and executed.

3.3.4.1. Generation of Multimodal Patches

Much like in Subsection 3.2.1.1, it was then decided that the CNN patches should be generated through the bounding box of the candidate superpixel and its neighbors superpixels. The difference here is that the patches are generated in pairs, one from FLAIR superpixels and other from T1 superpixels. This was possible because the FLAIR and T1 MRI were previously pre-processed, registered and shares the same superpixels grid.

Such as in Subsection 3.2.1.1, the patches dimensions are also corrected to be suitable to CNN. The procedure to generate CNN patches is illustrated by Figure 22.

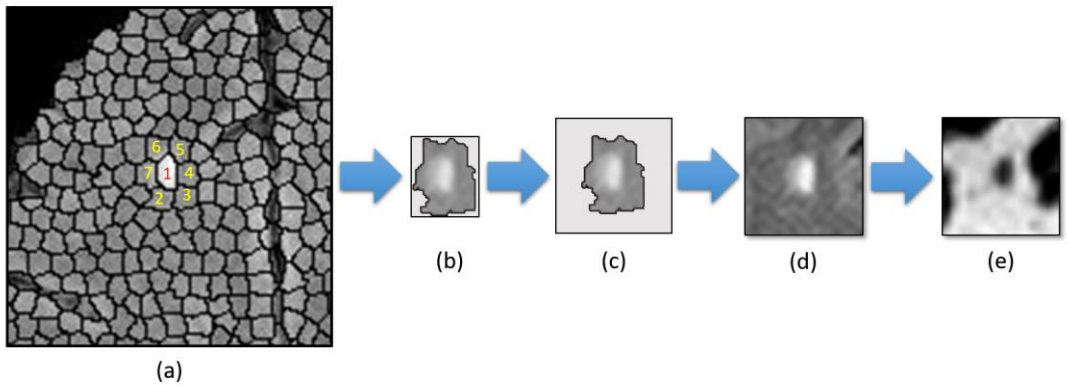


Figure 22: Generation of multimodal patches. (a) represents the superpixels of a slice of the brain in FLAIR MRI. The superpixel marked by the number 1 is the one you want to generate an patch on CNN. The superpixels marked by numbers 2 to 7 are their neighbors. (b) represents the bounding box of the superpixels marked by numbers 1 to 7 in (a). (c) represents the bounding box corrected using the largest dimensions of all bounding boxes from superpixels neighborhoods in database. (d) represents the patch generated from the pixels inside of the corrected bounding box. (e) represents the correspondent patch in a slice from the preprocessed T1 MRI which is aligned with FLAIR and shares the same superpixels grid.

3.3.4.2. Parallel CNN Train and Test

For the training and testing, the patches need to be divided into two groups: those with which they represent superpixels of WML and those representing superpixels of non-WML. This procedure is done using the markings like shown in Subsection 3.1.4.2.

After generating all the patches of all patients, it is now necessary to separate the training and validation databases for the training step and training and test databases for the test step. For each of these databases, patients MRI were selected randomly and patches from their superpixels were generated. However, for validation and test databases, only MRI marked by the specialist were used.

The training and test steps are done through a parallel CNN like in Subsection 3.2.1.2, but that uses multimodal patches: one from FLAIR MRI and other from T1 MRI. The idea is to extract features from both patches separately but combine then for more robust training.

Training sessions were executed to find the best parallel CNN parameters. All training sessions, however, shared some parameters. It was used 100 epochs as a stopping criterion, learning rate value 0.001, momentum value 0.9 and input batch of 64 images.

The architecture that resulted in the best model is as follows. Each path of convolution and downsampling layers consists of the following sequence of layers: (1) three convolutional layers, (2) one pooling layer, (3) other three convolutional layer, (4) other pooling layer. The outputs layers of these paths are merged and flattened to be the input layer of the MLP. MLP is composed of the following layers: (5) one input layer, resulting from the output of (4) merged and flattened in a 1D vector, (6) one hidden layer and (7) an output layer.

Knowing the sequence of layers of this network, their internal configuration are as follow: (1) use 20 filters of size 3 × 3 and ReLU activation function; (2) uses maxpooling function of size 2×2 and ReLU activation function, (3) use 50 filters of size 3×3 and ReLU activation function; (4) uses maxpooling function of size 2×2 and ReLU activation function; (5) connects with (6) using a dropout function with 0.5 chance of activation; (6) contains 500 neurons and ReLU activation function and (7) contains two neurons and uses a softmax function giving the probability of the input to belong to either class.

Figure 23 illustrates the architecture of this parallel CNN.

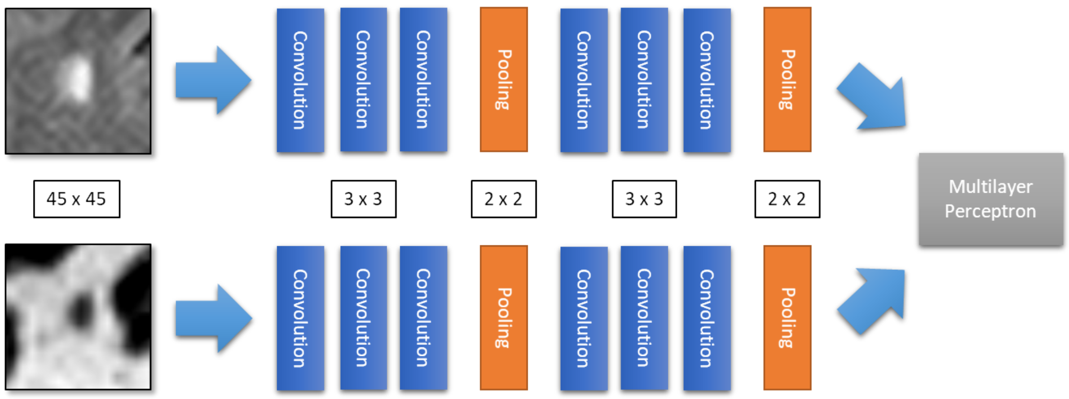


Figure 23: Architecture of the parallel CNN with multimodal patches.

3.4. Method 4

The fourth proposed method for WML detection uses pairs of FLAIR and T1 MRI. It is divided into four steps as described in Figure 24. The first three steps are the same as the Method 3 (Section 3.3), but here we use common CNNs with patches resulting from pixelwise comparison operations between correspondent FLAIR and T1 patches. Following, this fourth step of this method is explained in more details.

3.4.1. Candidates Classification

After segmentation of the candidates, the next step is to classify them as WML regions or non-WML regions using CNN. Following, it is explained in how the CNN entries were created and how the training and testing procedures were designed and executed.

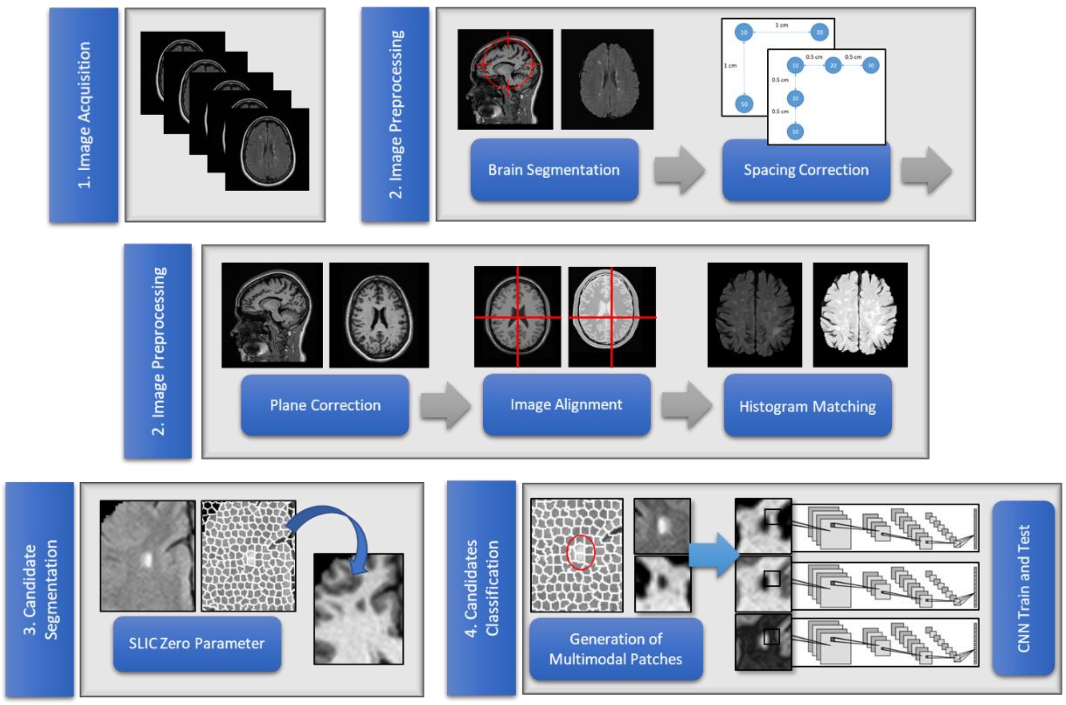


Figure 24: Method using T1 and FLAIR MRI.

3.4.1.1. Generation of Multimodal Patches

Much like in Subsection 3.3.4.1, for each candidate it is generated a pair of patches from T1 and FLAIR MRI, since they are pre-processed, aligned and shares the same superpixels grid. However, these patches are not used together in a parallel CNN, like in section 3.3.4.1. Here, the pairs of patches are merged through pixelwise comparison operations to create new patches with combined information.

Pixelwise comparison operations are operations performed at the pixel level. The operations used in this thesis are derived from the analysis of changes in aerial images for remote sensing [139], which have already been used effectively in medical imaging as well [140]. Therefore, this set was adapted to identify intensity differences in T1 and FLAIR MRI. From the operations described in [139], we used difference between images, image ratio and fuzzy XOR.

The operation of difference between images aims to measure differences of pixels between two images by subtracting them. This operation is described by Equation 22.

$$I(x,y) = I_F(x,y) - I_T(x,y) \tag{22}$$

where:

- $I_F(x,y)$ is the pixel value of FLAIR MRI in coordinates x and y;
- $I_T(x,y)$ is the pixel value of T1 MRI in coordinates x and y;
- $I(x,y)$ is the pixel value resulting from difference between images;

Like operation of difference between images, the operation of image ratio aims to measures differences between the pixels, but it uses the phase of the ratio between them. The phase indicates how similar the corresponding pixels are between the two images. This operation is described by Equation 23.

$$I(x,y) = \left|\arctan\left(\frac{I_F(x,y)}{I_F(x,y)}\right)\right| \tag{23}$$

where $I(x,y)$ is the pixel value resulting from image ratio. Note that arctan is the arctangent function returning values in degrees, so that the more $I_T(x,y)$ and $I_F(x,y)$ are dissimilar, the more arctangent value is next to 180 or -180 degrees, and de more similar they are, the more the arctangent value is close to 0. Since we are using absolute arctangent, the dissimilar values are next to 180 and similar next to 0. To avoid losing much information when converting the float degree values to pixel values for resulting image, we rescaled the values to interval 0 and the maximum value in DICOM standard.

The fuzzy XOR operation uses the concept of the exclusive OR operator of fuzzy logic to transform the pixel value of images. The result of this method highlights different regions and makes the similarities less obvious. Equation 24 shows how to calculate the fuzzy XOR operator.

$$I(x,y) = \max\left(\bar{I}_F(x,y) * I_T(x,y), \bar{I}_T(x,y) * I_F(x,y)\right) \tag{24}$$

where $I(x,y)$ is the pixel value of the image resulting from the fuzzy XOR operation and $\bar{I}_a(x,y)$ is a pixel value from some image a subtracted from value 1. In fuzzy XOR operation, the more different the pair of pixels is, the closer to 1 will be; and the more similar the pair of voxels, the closer to 0 will be its value. Note that before operation the values from T1 and FLAIR MRI are rescaled to interval of float values [0,1] and values resulting of the operation are rescaled to interval 0 and the maximum value in DICOM standard.

Figure 25 illustrates how patches resulting from pixelwise comparison operations are generated.

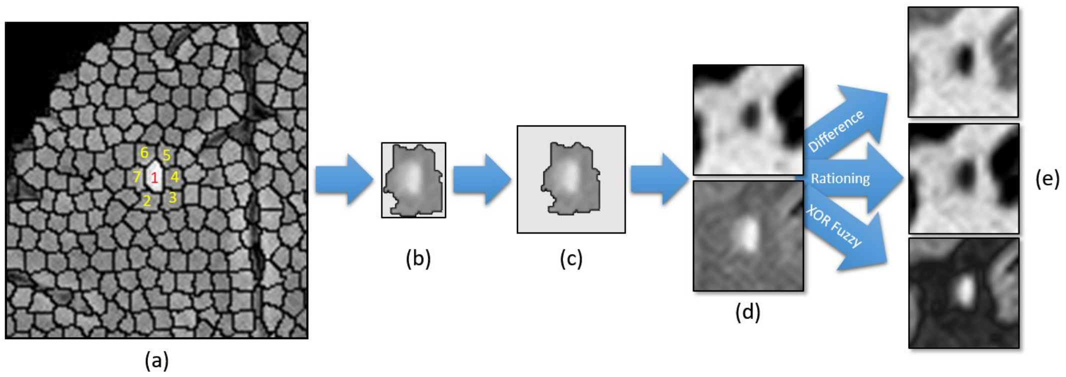


Figure 25: Generation of comparing patches. (a) represents the superpixels over a slice of a FLAIR MRI. The superpixel marked by the number 1 is the one you want to generate a patch. The superpixels marked by numbers 2 to 7 are its neighbors. (b) represents the bounding box of the superpixels marked by numbers 1 to 7 in (a). (c) represents the bounding box corrected using the largest dimensions of all bounding boxes from superpixels neighborhoods in database. (d) represents the patch generated from the pixels inside of the corrected bounding box and its correspondent in T1 MRI. (e) represents patches generated from pixelwise comparing operations between T1 and FLAIR patches from (d).

3.4.1.2. CNN Train and Test

After generating the patches for each superpixel, it is now necessary to separate patches as patches WML and non-WML regions and define the databases for training and test steps. This is all done such as Subsection 3.1.4.2, but this time there are three patches resulting from pixelwise comparison operations for each superpixel, as explained in Subsection 3.4.1.1.

Each of these types of patches are submitted in an independent CNN. So, this time, there are three CNNs executing separately, one for patches resulting from difference of images operation, other one for patches resulting from image ratio operation, and the last one for patches resulting from fuzzy XOR operation.

The architecture and training parameters of these three CNNs are the same the CNN from Method 1 (Subsection 3.1.4.2). They are illustrated in Figure 26.

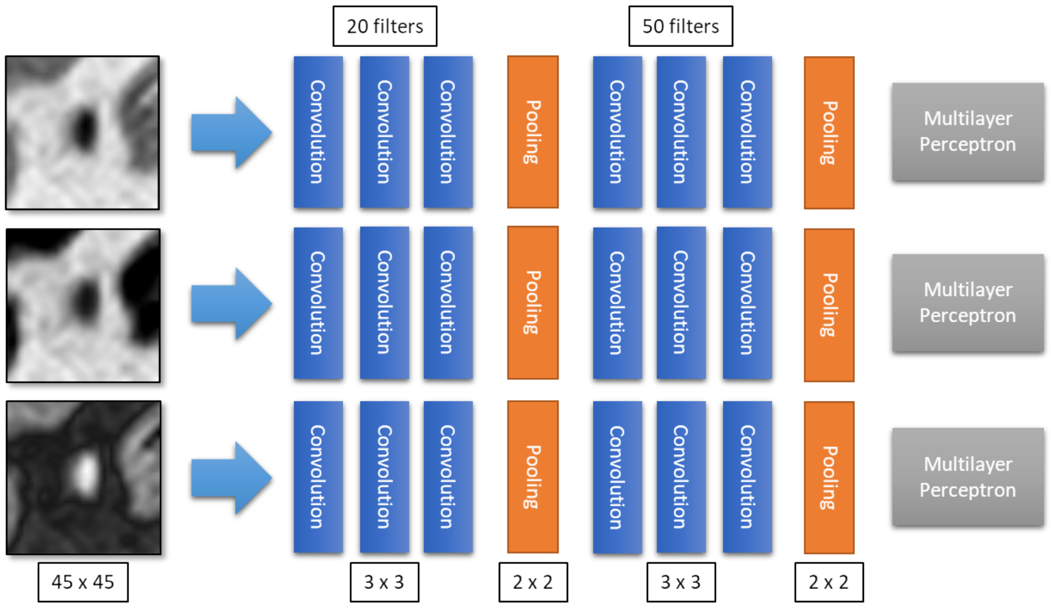


Figure 26: Architecture of CNNs with comparing patches.

4. Results and discussion

This Chapter will show the results obtained by applying the methods on the databases. These results will be discussed for a better understanding and compared with related works.

4.1. Train, validation and test images

First, it is necessary to understand how the MRI databases, specified in Subsections 3.1.1 and 3.3.1, were used. As explained in Sections 3.1, 3.2, 3.3 and 3.4, on each method, the images of these exams were preprocessed, had candidates for WML regions computed and, for each candidate, CNN patches were generated.

As explained in Subsections 3.1.4.1, 3.2.1.1, 3.3.4.1 and 3.4.1.1, the superpixels are divided into WML regions and non-WML regions using markings. Each WML marking is represented by one or more WML regions. Table 1 shows the relationship between the number of WML and WML regions from each of the 91 patients of the first database. Table 2 shows the relationship between the number of WML and WML regions from each of the 73 patients of the second database.

Next, the patches for WML and non-WML regions are generated and used in the training and test steps. For the first database, the exams were divided into three groups: 68 patients were separated for the training group, 10 for the validation group of the training and 15 for the test group. For the second database, the three groups are: 53 patients for the training group, 10 for the validation group of the training and 10 for the test group. Patients from each group were randomly selected, but for test and validation groups, only exams marked by specialist were used.

In the training phase, it is used the patches from the training and validation groups. Considering patches of training group, it was decided to use only 10% of the patches of non-WML regions. This choice was done to reduce the computational time of training phase and for avoiding the unbalanced training, since there are much more non-WML regions than WML regions in MRI.

The choice of which non-WML regions are used for training was done carefully, assuring that regions of whole brain were well represented in the training. This selection was done as follow: superpixels of all brain tissue were labeled such as they were sorted first from left to right and next from top to bottom. Once these superpixels are sorted, for each sequence of 10 superpixels of non-WML regions, the first one is maintained and the following nine are disregarded. The maintained superpixels representing non-WML regions are those whose patches were selected for training group.

Considering the patches generated from validation group, all images of lesion regions and non-lesion regions were selected.

Table 1: Number of WML and regions of WML per patient in database 1.

Patient	**WML**	**WML region**	**Patient**	**WML**	**WML region**	**Patient**	**WML**	**WML region**
1	55	228	32	43	290	63	11	23
2	9	17	33	40	241	64	3	4
3	16	27	34	31	75	65	54	81
4	26	47	35	106	544	66	41	88
5	3	3	36	5	12	67	11	73
6	114	336	37	35	168	68	6	14
7	3	6	38	25	133	69	24	46
8	6	13	39	179	1523	70	47	113
9	112	439	40	6	16	71	43	108
10	24	89	41	45	186	72	5	6
11	122	360	42	87	218	73	23	41
12	4	8	43	21	87	74	11	13
13	27	43	44	91	329	75	89	333
14	91	188	45	125	258	76	10	17
15	36	206	46	179	406	77	179	437
16	133	594	47	50	220	78	1	1
17	18	96	48	9	11	79	32	219
18	51	153	49	46	76	80	72	225
19	7	23	50	21	26	81	36	334
20	39	132	51	210	931	82	28	104
21	69	129	52	4	6	83	7	32
22	84	287	53	7	17	84	94	655
23	203	903	54	34	71	85	14	106
24	136	628	55	26	48	86	16	128
25	7	11	56	33	86	87	10	42
26	17	90	57	37	124	88	59	216
27	180	646	58	11	27	89	42	435
28	12	56	59	29	65	90	15	31
29	110	268	60	140	407	91	50	716
30	16	87	61	34	65	92	9	18
31	1	1	62	11	17	93	6	41

Table 3 shows the number of patients, WML, WML regions and non-WML regions from the training and validation groups in database 1. Table 4 shows the same information, but for database 2.

Table 2: Number of WML and regions of WML per patient in database 2.

Patient	WML	WML region	Patient	WML	WML region	Patient	WML	WML region
1	9	18	26	43	289	51	11	14
2	16	26	27	40	226	52	11	23
3	26	46	28	31	75	53	3	4
4	3	3	29	106	552	54	54	79
5	114	318	30	5	12	55	11	73
6	3	6	31	35	158	56	42	106
7	6	14	32	25	132	57	5	6
8	112	433	33	177	1512	58	23	43
9	122	363	34	6	16	59	11	14
10	4	8	35	45	190	60	89	330
11	27	43	36	86	227	61	10	19
12	91	182	37	124	268	62	179	438
13	133	598	38	178	415	63	1	1
14	18	97	39	50	228	64	55	227
15	7	24	40	9	13	65	24	87
16	39	130	41	21	26	66	36	198
17	69	136	42	4	5	67	204	899
18	84	293	43	7	17	68	110	269
19	136	653	44	34	71	69	21	82
20	7	9	45	26	49	70	91	329
21	17	92	46	34	72	71	208	951
22	180	650	47	11	27	72	38	129
23	12	56	48	29	61	73	41	88
24	16	87	49	141	407			
25	1	1	50	34	61			

Table 3: Characteristics of the groups of images of the training step in database 1.

Training step	Train group	Validation group
Number of patients	68	10
Number of WML	3181	825
Number of WML regions	10942	3246
Number of non-WML regions	120104	171356

Table 4: Characteristics of the groups of images of the training step in database 2.

Training step	Train group	Validation group
Number of patients	53	10
Number of WML	1938	1065
Number of WML regions	5933	4612
Number of non-WML regions	89459	183474

In the test step, the patches generated from the training and validation groups are merged into a single group, resultingww in a new training group composed of 78 patients for database 1 and 63 patients for database 2. This group obeys the same criteria for selecting patches from the training group from training step.

Considering the patches generated from the test group, all WML and non-WML regions images were selected.

Table 5 shows the number of patients, WML, regions of WML and regions of non-WML of the training and test groups of the test step. Table 6 shows the same information, but for database 2.

Table 5: Characteristics of the groups of images of the test step in database 1.

Test step	Train group	Test group
Number of patients	78	15
Number of WML	4009	490
Number of WML regions	14195	3302
Number of non-WML regions	137123	275580

Table 6: Characteristics of the groups of images of the test step in database 2.

Test step	Train group	Test group
Number of patients	63	10
Number of WML	3003	828
Number of WML regions	10545	3259
Number of non-WML regions	107615	164937

4.2. Training step

Training and validation patches were used to search for better CNN settings in each method. For this, training sessions were performed to compare various configurations of CNN. At the end of each training, the generated models were stored. Then, these models were used to classify patches of the validation group. The results obtained in each model were compared and the model that obtained the best results was chosen.

Table 7 shows the validation metrics obtained by applying the best model in the validation database for each method. In this table, Method 1, 2 and 3 represents the methods explained in Sections 3.1, 3.2 and 3.3, respectively. Methods 4.1, 4.2 and 4.3 represents the operations difference of images, image ratio and fuzzy XOR explained in Section 3.4.

Table 7: Validation metrics of the training step.

Train step	Method 1	Method 2	Method 3	Method 4.1	Method 4.2	Method 4.3
Sensitivity	80,60%	79,51%	74,81%	71,32%	69,27%	65,90%
Specificity	97,61%	97,97%	96,82%	88,55%	87,67%	93,02%
Accuracy	96,19%	96,43%	95,45%	87,48%	88,40%	91,33%

4.3. Test step

Once the best CNN settings have been defined, they can be used to classify the test patches.

For generation of the test models for each method, the training and validation groups were joined in a new training group. These new groups are trained again using the best CNN settings found in the training step. The generated models of this procedure are then used to classify the test patches.

Table 8 shows the results obtained by applying the model from each method in their respective test groups.

Table 8: Validation metrics of the test step.

Test step	Method 1	Method 2	Method 3	Method 4.1	Method 4.2	Method 4.3
Sensitivity	90,12%	89,85%	76,03%	74,62%	74,31%	67,56%
Specificity	98,02%	97,91%	97,82%	89,85%	85,54%	90,03%
Accuracy	97,93%	97,82%	97,40%	89,55%	85,33%	89,60%

4.4. Discussion

The results show the efficiency of the proposed methods, even in a heterogeneous database, composed by exams from different MRI devices and WML related to different pathologies. The parameter values used are essential to achieve these results.

First, as explained in Section 3.1.3 and 3.3.3, it was decided to use SLIC0 with 3500 superpixels per slice. This value was empirically the best choice to possibility detection of even the smaller lesions but also allows that larger lesions could be formed by sets of superpixels.

Also, as explained in Subsection 2.4.2, it was decided that CNN patches, used to represent superpixels, should be of size 45x45 for Methods 1, 3 and 4, and 42x42 for Method 2. This possibilities that patches could be large enough to carry meaningful information and also possibilities to capture useful information from tissues surrounding the lesions.

Finally, as explained in Subsection 2.4.2, the parameters from CNN were empirically chosen based on training sessions. Their choice, however, were influenced by the size of patches because if patches were applied in many layers of convolution and pooling, it could result in few pixels or none to reach the fully connected layer, making it impossible for the network to distinguish the classes between WML region and non-WML region.

On the other hand, if fewer convolution and pooling layers were used, there would be fewer and less representative feature maps, and consequently less features would be presented to the fully connected layer. This would generate few individual characteristics for each patch and proportionally more pixels, making the network less discriminating. In addition, our approach also generated less false positives in detriment of the correctness of the lesion regions.

The first convolutional layers of a CNN typically detect various forms of edges, corners and other basic structuring elements [109], which is not important in this case, since edges in white matter lesions and its regions do not tend to be well-defined. Thus, for the proposed methods, networks with more convolution layers obtained the best results.

As important as the chosen parameters are to achieve these results, they expose the limitations of their algorithms.

First, it is known that one of the limitations of CNN is that it commonly requires a larger number of patches to achieve a good model. This limitation is reduced in Methods 1 and 2 since the database 1 has a reasonable size. Also, since it is used regions in all brain, there are many patches to train CNN. However, the Methods 3 and 4 achieved lower results, even using T1 and FLAIR patches, because of the size of database 2, which generated much less patches for an effective training.

Other limitation of the proposed methods is that, while using small size superpixels assure that smaller lesions are well represented, the larger ones, which are composed by many superpixels, does not always has all of it superpixels detected. So, this is also a limitation in detecting lesions related to different types of pathologies.

A limitation in CNN is the number of parameters. A reasonable CNN requires a great number of parameters since it normally presents input layers, convolution layers, pooling layers and the fully connected layer and more. So, the study of these parameters commonly requires a lot of time and experimenting. To overcome that, in this work, many training sessions were used to search for the best parameters.

Finally, CNN is also known for its demand for high computational time. This also increases as the number of patches increases. In this work, where many training sessions were necessary to search for the best configuration for CNN, this became evident. The obvious solution is to use a powerful hardware. For training sessions, it was used one NVIDIA Titan X graphics cards to possibility faster results. On the other hand, after the training phase is completed, the classification of all patches from a MRI brain is done in some seconds, even using CPU. For example, applying all steps of Method 1 for classification in one complete MRI takes around just three minutes in an Intel Core i5 processor of 2.8 ghz.

4.5. Study of cases

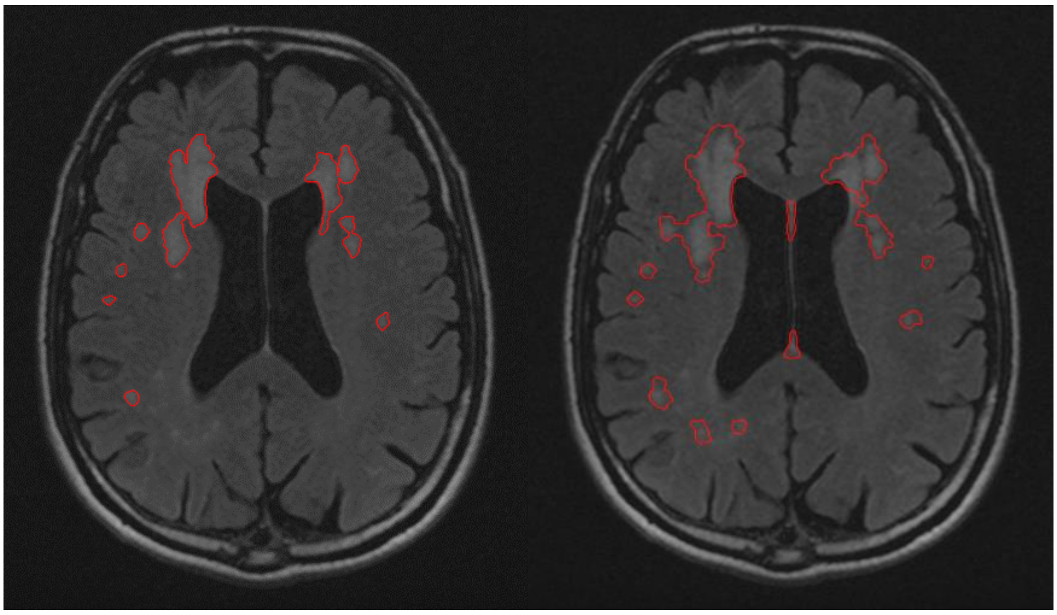
In this section, it is presented some study of cases. They illustrate the results of the detection over MRI slices and compare them with their respective markings. Each study of case is from one slice of different patients. The study of cases are related to Method 1 (Section 3.1).

The Figure 27 shows that the method could detect almost all WML. It also shows that it was able to segment almost as precise as the marking.

The Figure 28 show that almost all WML were detected. However, this result also shows a number of false positives. This has happened probably because there are regions of hyperintensity that are not WML. It is expected that training with more cases could reduce such cases.

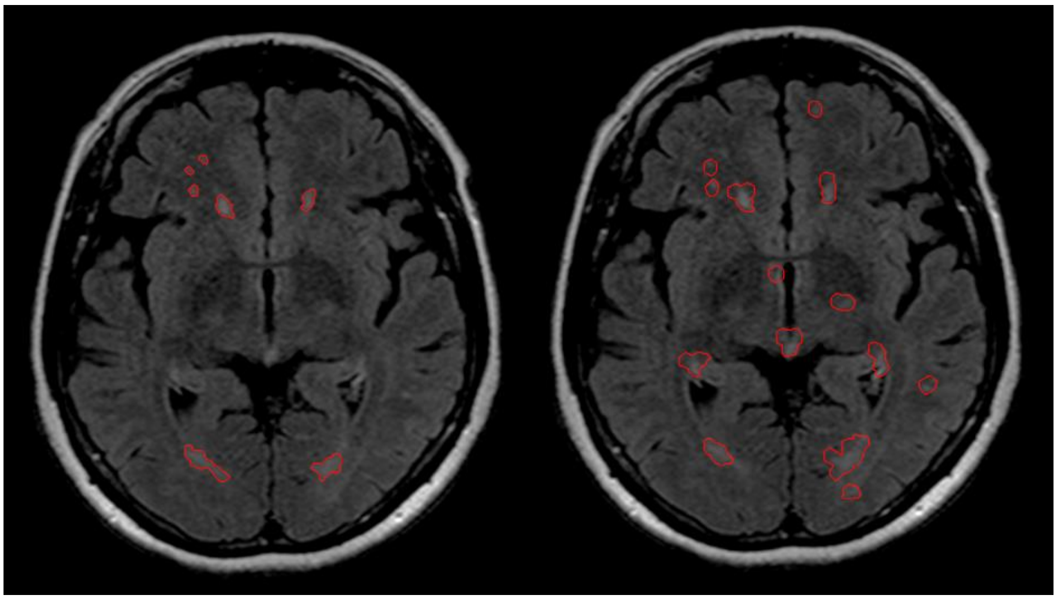
Figure 29 shows other case where all WML were detected. As we can see, a false positive was also found. However, this false positive may actually be a WML with less than 1 mm of diameter. Specialist was instructed to ignore such cases, but it is interesting to note that our method was able to find it.

Figure 30 shows other case where there all WML were detected, but with false positives. Some of these false positives are actually WML that were ignored by its small sizes, but other are real false positives. Again, it’s expected that a larger train database can overcome these issues.



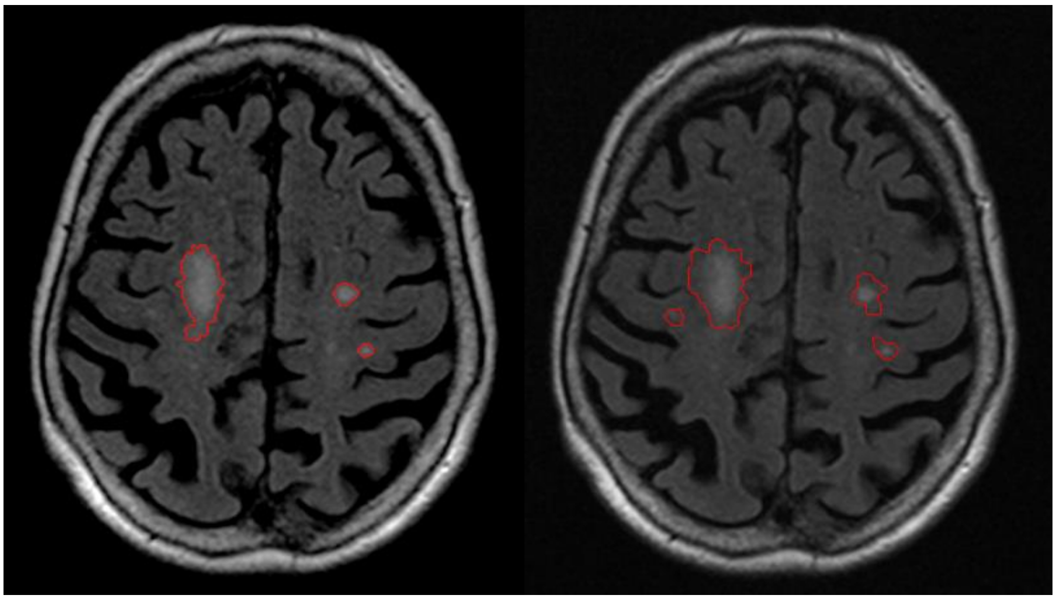
(a) (b)

Figure 27: Study of case 1. (a) represents a slice with marked WML and (b) represents the detection result.



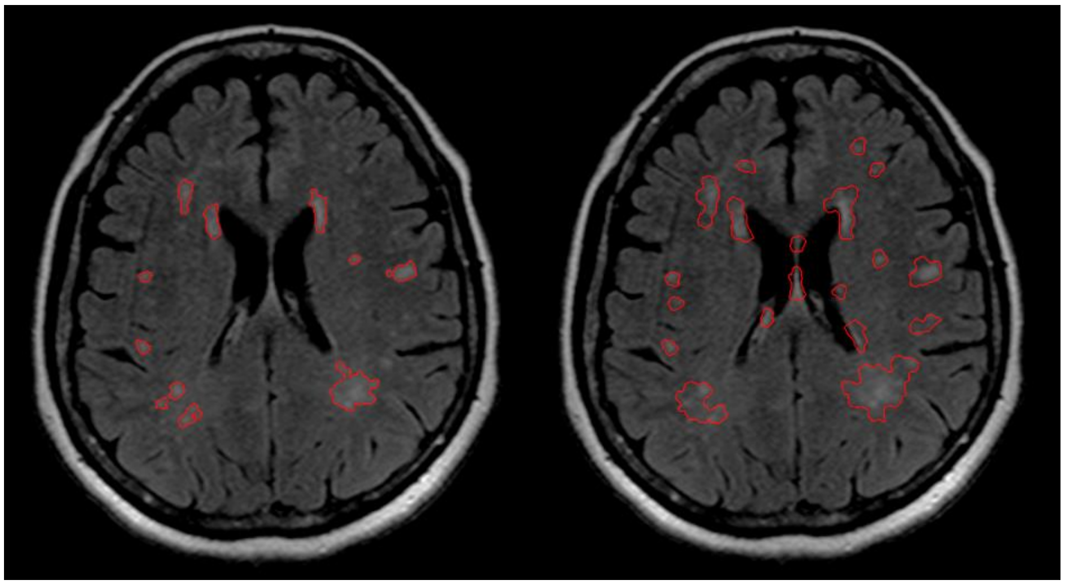
(a) (b)

Figure 28: Study of case 2. (a) represents a slice with marked WML and (b) represents the detection result.



(a) (b)

Figure 29: Study of case 3. (a) represents a slice with marked WML and (b) represents the detection result.



(a) (b)

Figure 30: Study of case 4. (a) represents a slice with marked WML and (b) represents the detection result.

4.6. Comparison with related works

In this section, it is presented a comparison of the results of this work with other related works. These related works are shown in Table 9, that presents information about types of diseases associated to the lesions, number of patients composing the databases and performance metrics (sensitivity and specificity) about their methods.

Table 9: Comparison with related works

Work	Disease	Number of Subjects	Sensitivity	Specificity
Anbeek et al. [110]	Arterial vascular disease	20	97%	97%
Lao et al. [111]	Diabetes	45	85%	99%
Kruggel et al. [112]	Dementia	116	90%	91%
de Boer et al. [113]	Different types	6	79%	—
Khademi et al. [114]	Different types	24	82%	99%
Wang et al. [115]	Different types	272	81%	97%
Ong et al. [116]	Different types	61	67%	40%
Shi et al. [117]	Acute Infarction	91	80%	—
Roura et al. [8]	Lupus	30	62%	80%
Method 1	Different types	91	90,12%	98,02%
Method 2	Different types	91	89,85%	97.91%
Method 3	Different types	73	76,03%	97,82%
Method 4.1	Different types	73	74,62%	89,85%
Method 4.2	Different types	73	74,31%	85.84%
Method 4.3	Different types	73	67,56%	90.03%

Anbeek et al presents arguably the highest results from related works. However, the database used in his work is too small, which makes difficult to validate it and evaluate the generality of its method. In addition, it only considers WML of arterial vascular disease. Lao et al., Kruggel et al., Shi et al. and Roura et al. works also consider only lesions related to a single disease. In fact, it is not common to find works that detect lesions related to several brain pathologies in literature.

de Boer et al. and Khademi et al. works considers WML of different types of diseases, but their databases have few samples, which compromises their evaluation. On the other hand, Shi et al. presents an average size database and considers several types of diseases, but it presents lower sensitivity and specificity than this proposed work. The only work that presents a well populated database, considers different types of diseases and present competitive results is Wang et al.

The comparison with related works shows that the proposed method has significant results compared with literature. Although it is not common works that detect WML from different diseases, this work presents high results compared with those who does. The first two proposed methods show excellent results compared with the related works while other proposed methods, with fewer samples to train, has potential to achieve even better results once we acquire a larger database, since they use FLAIR and T1 MRI together.

It is worth remembering that it is not possible to make a faithful comparison with the related works, since the related works do not use the same database of this method. However, the database of this thesis is quite heterogeneous, since images were acquired from different MRI devices and present lesions of different types of diseases, which shows that this method can present robust results regardless of the materials used in it.

The proposed method presents other advantages to the related works.

First, it is important to note that proposed method does not rely on any candidates reduction technique. Here, all regions of the brain, subdivided in superpixels, are classified by machine learning, so there are much more candidates to classify than the related works. This shows that the presented method is reliable to classify regions of all the brain tissue.

Also, CNN does not need an explicit feature extraction step, as explained in Subsection 2.4.1., which is not a simple task. Finally, since SLIC0 contains only one parameter and segment entire image with good results and CNN does not rely on a feature extraction step, that is normally specific each problem, and classifies all superpixels, the proposed method can be considered as generalist and has a large potential to be used in other modalities of images and problems.

Thus, it is concluded that the proposed method occupies a prominent place in the literature, being able to reach competitive results in the task of WML detection, especially if compared with methods that detects WML related with different types of diseases.

5. Conclusions

5.1. Work Overview

This work presented four methodologies for the detection of WML regions in MRI.

The first method can be applied on MRI of modality FLAIR. It uses preprocessing techniques for image enhancement, SLIC0 for candidates segmentation and CNN for candidates classification.

The second method also can be applied on MRI of modality FLAIR. The main difference from first method is that it uses parallel CNN and multiscale patches for candidates classification.

The third method can be used in pairs of MRI, one of modality T1 and other of modality FLAIR. It uses preprocessing techniques for image enhancement and image registration for better combined usage of the images. Then, it uses SLIC0 for candidates segmentation on both images and proposes a parallel CNN for candidates classification.

The fourth method also can be used in pairs of MRI, one of modality T1 and other of modality FLAIR. The main difference from third method is that it uses common CNNs with patches generated from pixelwise comparing operations applied in T1 and FLAIR images.

To validate the proposed methods, a private, heterogeneous and well-populated database was used in order to detect WML regions.

5.2. Work Evaluation

The first and second method has promising results while the third and fourth depends on a larger database for achieve presumably even better results.

The use of preprocessing techniques helped to greatly reduces the images area of study, maintaining only the brain. The results show that the brain segmentation worked well in all the database.

The use of histogram matching, which is commonly used in databases to reduce inhomogeneity issues among the images also improved the results [100–102].

In general, the use of SLIC0 proved to be very promising on the generation of CNN patches. However, since there are too much candidates on each MRI, it has been exposed the possible need for candidate reduction techniques. The idea is to discard the excess of regions before train and test steps.

It has also been proposed to use CNN for classification task. The CNN showed to be quite effective. Even with simple architectures, it was possible to detect about a great number of WML regions.

We also highlight the large number of empirical parameters used on CNN, which may become a negative factor. On the other hand, it is noteworthy that using a CNN, an explicit extraction and feature selection step is unnecessary, which in machine learning is usually a disturbing step.

The detection of WML regions also presents significant results compared to related works. In addition, many studies investigate only one type of pathology while our work has excellent results in detection of different types of WML. Thus, a robust method capable of generalizing the detection of WML in the white matter of the brain is presented.

Concluding, the method presented is very promising. It is possible to affirm that the methodologies may assist the specialists on the task of WML detection.

5.3. Work Contributions

This thesis presents many contributions. Here are those considered more relevant.

The main contributions of this work are its own innovative methods. Their innovation comes from the usage of CNN and parallel CNN for classifying SLIC0 superpixels-based patches, which was never done before. Such innovations bring good segmentation results while being, in theory, much faster than other methods in literature.

Other main contribution is a robust and user-friendly segmentation software (Appendix A), which contains many visualization and segmentation tools and can be used not only for Brain MRI but also to any image in DICOM standard. It also features an automatic WML segmentation based on the first method of this thesis (Section 3.1).

The last main contribution is a large database composed by more than 500 MRI of modality FLAIR with specialist markings that is being constructed with the use of the segmentation software. This database is still being marked and will be available public soon.

5.4. Future Works

Even with good results, there is still a good number of changes that can be applied to achieve higher performance in the results. Following, it is presented some method issues and proposals to overcome them as future works.

First, since we are classifying all candidates generated from SLIC0 superpixels, it would be interesting trying to implement some candidate reduction technique. Once many superpixels are from white matter hypointense regions, which are normally healthy tissue, unsupervised clustering techniques based only on image intensity, such as K-means and Fuzzy C-means, could have good results. Also, trying to remove gray matter, which have hyperintense tissue like WML tissue, could improve the results.

One idea to possible improvement in candidates segmentation is to use SLIC0 with different sizes of superpixels, once WML appears in different sizes.

Other future work could be an analysis to improve the architecture of CNN and, consequently, the classification results. It would be interesting to use techniques in order to find the network parameters automatically, such as evolutionary and other optimization algorithms.

Although the CNN presented good results, there are other machine learning techniques that could be used. Support Vector Machines (SVM) were widely used on many machine learning problems and even in WML detection [68-70]. So, SVM with different kinds of attributes (including texture) could be an alternative to CNN.

Finally, since the candidates are superpixels, the contour of final segmentation may be less effective than a pixelwise classification. So, one future work can be a second classification focused on pixels inside superpixels. This would refine the contour of the segmentation while being much faster than a pixelwise classification of all brain pixels.

6. References

1. DING, J. R. et al. **Abnormal functional connectivity density in patients with ischemic white matter lesions**: An observational study. Medicine. 95 (36). 2016.

2. SUDRE, C. H. et al. **Longitudinal segmentation of age-related white matter hyperintensities.** Medical Image Analysis. 38: 50–64. 2017.

3. BRUN, A.; ENGLUND, E. **A white matter disorder in dementia of the Alzheimer type**: a pathoanatomical study. Annals of neurology. 19 (3): 253–262. 1986.

4. FIELDS, R. D. **White matter in learning, cognition and psychiatric disorders**. Trends in neurosciences. 31 (7): 361–370. 2008.

5. CASEY, B.; GIEDD, J. N.; THOMAS, K. M. **Structural and functional brain development and its relation to cognitive development**. Biological psychology. 54 (1): 241–257. 2000.

6. LISTON, C. et al, **Frontostriatal microstructure modulates efficient recruitment of cognitive control**. Cerebral Cortex. 16 (4): 553–560. 2006.

7. WALHOVD, K. B.; FJELL, A. M. **White matter volume predicts reaction time instability**. Neuropsychologia. 45 (10): 2277–2284. 2007.

8. BENGTSSON, S. L. et al. **Extensive piano practicing has regionally specific effects on white matter development**. Nature neuroscience. 8 (9): 1148–1150. 2005.

9. ROURA, E. et al. **Automated detection of lupus white matter lesions in MRI.** Frontiers in neuroinformatics. 10. 2016.

10. GHAFOORIAN, M. et al. **Location sensitive deep convolutional neural networks for segmentation of white matter hyperintensities**. arXiv preprint arXiv:1610.04834. 2016.

11. JACK, C. R. et al. **FLAIR histogram segmentation for measurement of leukoaraiosis volume**. Journal of Magnetic Resonance Imaging. 14 (6): 668–676. 2001.

12. HALLER, S. et al. **Do brain T2/FLAIR white matter hyperintensities correspond to myelin loss in normal aging?** A radiologic-neuropathologic correlation study. Acta neuropathologica communications. 1 (1): 14. 2013.

13. HACHINSKI, V. C.; POTTER, P.; MERSKEY, H. **Leuko-araiosis**, Archives of neurology. 44 (1): 21–23. 1987.

14. VELDINK, J. H. et al. **Progression of cerebral white matter hyperintensities on MRI is related to diastolic blood pressure.** Neurology. 51 (1): 319–320. 1998.

15. SCHMIDT, R. et al. **Progression of cerebral white matter lesions**: 6-year results of the Austrian Stroke Prevention Study. The Lancet. 361 (9374): 2046–2048. 2003.

16. PODEWILS, L. et al. **Physical activity and white matter lesion progression Assessment using MRI.** Neurology. 68 (15): 1223–1226. 2007.

17. GOUW, A. A. et al. **Progression of white matter hyperintensities and incidence of new lacunes over a 3-year period**. Stroke. 39 (5): 1414– 1420. 2008.

18. MOK, V. C. et al. **Effects of statins on the progression of cerebral white matter lesion**, Journal of neurology. 256 (5): 750–757. 2009.

19. PANTONI, L.; GARCIA, J. H. **The significance of cerebral white matter abnormalities 100 years after Binswanger's report**. Stroke. 26 (7): 1293–1301. 1995.

20. YLIKOSKI, A. et al. **White matter hyperintensities on MRI in the neurologically nondiseased elderly**. Stroke. 26 (7): 1171–1177. 1995.

21. LIAO, D. et al. **Presence and severity of cerebral white matter lesions and hypertension, its treatment, and its control**. Stroke. 27 (12): 2262–2270. 1996.

22. LONGSTRETH, W. et al. **Clinical correlates of white matter findings on cranial magnetic resonance imaging of 3301 elderly people**. Stroke. 27 (8): 1274–1282. 1996.

23. DE LEEUW, F. et al. **Prevalence of cerebral white matter lesions in elderly people:** a population based magnetic resonance imaging study. The Rotterdam Scan Study. Journal of Neurology. Neurosurgery & Psychiatry. 70 (1): 9–14. 2001.

24. XIONG, Y. Y.; MOK, V. **Age-related white matter changes**. Journal of aging research. 2011.

25. BRETELER, M. et al. **Cerebral white matter lesions, vascular risk factors, and cognitive function in a population-based study The Rotterdam Study**. Neurology 44 (7): 1246–1246. 1994.

26. GARDE, E. et al. **Relation between age-related decline in intelligence and cerebral white matter hyperintensities in healthy octogenarians:** a longitudinal study. The Lancet. 356 (9230): 628–634. 2000.

27. LAUNER, L. J. **Epidemiology of white matter lesions**. Topics in Magnetic Resonance Imaging. 15 (6): 365–367. 2004.

28. LAUNER, L. J. et al. **Regional variability in the prevalence of cerebral white matter lesions:** an MRI study in 9 European countries (CASCADE). Neuroepidemiology. 26 (1): 23–29. 2005.

29. WEN, W. et al. **White matter hyperintensities in the forties:** their prevalence and topography in an epidemiological sample aged 44–48. Human brain mapping. 30 (4): 1155–1167. 2009.

30. DEBETTE, S.; MARKUS, H. **The clinical importance of white matter hyperintensities on brain magnetic resonance imaging:** systematic review and meta-analysis. Bmj. 341: c3666. 2010.

31. DAVENPORT, A. **High blood pressure and cerebral white matter lesion progression in the general population**. Kidney International. 84: 223–224. 2013.

32. VAN SWIETEN, J. et al. **Hypertension in the elderly is associated with white matter lesions and cognitive decline**. Annals of neurology. 30 (6): 825–830. 1991.

33. MANOLIO, T. A. et al. **Magnetic resonance abnormalities and cardiovascular disease in older adults**. The Cardiovascular Health Study. Stroke. 25 (2): 318–327. 1994.

34. LIAO, D. et al. **The prevalence and severity of white matter lesions, their relationship with age, ethnicity, gender, and cardiovascular disease risk factors**: the ARIC Study. Neuroepidemiology. 16 (3): 149–162. 1997.

35. BASILE, A. M. et al. **Age, hypertension, and lacunar stroke are the major determinants of the severity of age-related white matter changes**. Cerebrovascular diseases. 21 (5-6): 315–322. 2006.

36. CHOI, H. S. et al. **Cerebral white matter hyperintensity is mainly associated with hypertension among the components of metabolic syndrome in Koreans**. Clinical endocrinology. 71 (2): 184–188. 2009.

37. KULLER, L. H. et al. **Relationship of hypertension, blood pressure, and blood pressure control with white matter abnormalities in the Women's Health Initiative Memory Study (WHIMS)** — MRI trial. The Journal of Clinical Hypertension. 12 (3): 203–212. 2010.

38. FAZEKAS, F. et al. **White matter signal abnormalities in normal individuals:** correlation with carotid ultrasonography, cerebral blood flow measurements, and cerebrovascular risk factors. Stroke. 19 (10): 1285–1288. 1988.

39. O'BRIEN, J. T. **Vascular Cognitive Impairment**. The American journal of geriatric psychiatry. 14 (9): 724–733. 2006.

40. VAN NORDEN, A. G. et al. **Causes and consequences of cerebral small vessel disease**. The RUN DMC study: a prospective cohort study. Study rationale and protocol. BMC neurology. 11 (1): 29. 2011.

41. PRINS, N. D.; SCHELTENS, P. **White matter hyperintensities, cognitive impairment and dementia:** an update. Nature Reviews Neurology. 11 (3): 157–165. 2015.

42. SCHOONHEIM, M. M. et al. **Sex-specific extent and severity of white matter damage in multiple sclerosis:** Implications for cognitive decline, Human brain mapping. 35 (5): 2348–2358. 2014.

43. MARSHALL, G. et al. **White matter hyperintensities and cortical acetylcholinesterase activity in parkinsonian dementia**. Acta neurologica scandinavica. 113 (2): 87–91. 2006.

44. LEE, S. J. et al. **The severity of leukoaraiosis correlates with the clinical phenotype of Parkinson's disease**. Archives of gerontology and geriatrics. 49 (2): 255–259. 2009.

45. LAWEK, J. S. et al. **The influence of vascular risk factors and white matter hyperintensities on the degree of cognitive impairment in Parkinson's disease**. Neurologia i neurochirurgia polska. 42 (6): 505–512. 2007.

46. SOHN, Y. H.; KIM, J. S., **The Influence of White Matter Hyperintensities on the Clinical Features of Parkinson's Disease**. Yonsei medical journal. 39 (1): 50–55. 1998.

47. WEINSTEIN, G. et al. **Brain imaging and cognitive predictors of stroke and Alzheimer disease in the Framingham Heart Study**. Stroke. 44 (10): 2787–2794. 2013.

48. HÂNCU, A.; RASANU; I., BUTOI, G. **White matter changes in cerebrovascular disease:** Leukoaraiosis, Advances in Brain Imaging. Europe: InTech. 249–250. 2009.

49. FU, J. et al. **Extent of white matter lesions is related to acute subcortical infarcts and predicts further stroke risk in patients with first ever ischaemic stroke**. Journal of Neurology. Neurosurgery & Psychiatry. 76 (6): 793–796. 2005.

50. TANG, W. K. et al. **Frequency and determinants of poststroke dementia in Chinese**. Stroke. 35 (4): 930– 935. 2004.

51. JIMENEZ-CONDE, J. et al. **Hyperlipidemia and reduced white matter hyperintensity volume in patients with ischemic stroke**. Stroke. 41 (3): 437–442. 2010.

52. MANTYLA, R. et al. **The prevalence and distribution of white-matter changes on different MRI pulse sequences in a post-stroke cohort**. Neuroradiology. 41 (9): 657–665. 1999.

53. HIRONO, N. et al. **Impact of white matter changes on clinical manifestation of Alzheimer's disease**. Stroke. 31 (9): 2182–2188. 2000.

54. AHARON-PERETZ, J.; CUMMINGS, J. L.; HILL, M. A. **Vascular dementia and dementia of the Alzheimer type**: cognition, ventricular size, and leukoaraiosis. Archives of neurology. 45 (7): 719–721. 1988.

55. DE LEEUW, F. E.; BARKHOF, F.; SCHELTENS, P. **White matter lesions and hippocampal atrophy in Alzheimer's disease**. Neurology. 62 (2): 310–312. 2004.

56. TARGOSZ-GAJNIAK, M. et al. **Cerebral white matter lesions in patients with dementia–from MCI to severe Alzheimer's disease**. Journal of the neurological sciences. 283 (1): 79–82. 2009.

57. BRICKMAN, A. M. et al. **Long-term blood pressure fluctuation and cerebrovascular disease in an elderly cohort**. Archives of neurology. 67 (5): 564–569. 2010.

58. CARMELLI, D. et al. **Evidence for genetic variance in white matter hyperintensity volume in normal elderly male twins**. Stroke. 29 (6): 1177–1181. 1998.

59. ZHANG, Y.; BRADY, M.; SMITH, S. **Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm**. IEEE transactions on medical imaging. 20 (1): 45–57. 2001.

60. BITAR, R. et al. **MR pulse sequences:** what every radiologist wants to know but is afraid to ask 1. Radiographics. 26 (2): 513–537. 2006.

61. SIMON, J. et al. **Standardized MR imaging protocol for multiple sclerosis:** Consortium of MS Centers consensus guidelines. American Journal of Neuroradiology. 27 (2): 455–461. 2006.

62. SAAD, N. M. et al. **Review of brain lesion detection and classification using neuroimaging analysis techniques.** Jurnal Teknologi. 74 (6): 73–85. 2015.

63. DE COENE, B. et al. **MR of the brain using fluid-attenuated inversion recovery (FLAIR) pulse sequences**. American journal of neuroradiology. 13 (6): 1555–1564. 1992.

64. WARDLAW, J. M. et al. **Neuroimaging standards for research into small vessel disease and its contribution to ageing and neurodegeneration**. The Lancet Neurology. 12 (8): 822–838. 2013.

65. GARCÍA-LORENZO, D. et al. **Review of automatic segmentation methods of multiple sclerosis white matter lesions on conventional magnetic resonance imaging**. Medical image analysis. 17 (1): 1–18. 2013.

66. KLOPPEL, S. et al. **A comparison of different automated methods for the detection of white matter lesions in MRI data**. NeuroImage. 57 (2): 416–422. 2011.

67. KAMNITSAS, K. et al. **Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation**. Medical image analysis. 36: 61–78. 2017.

68. SELVANAYAKI, K.; KARNAN, M. **CAD system for automatic detection of brain tumor through magnetic resonance image** - a review,

International Journal of Engineering Science and Technology. 2 (10): 5890–5901. 2010.

69. PADILLA, P. et al. **NMF-SVM based CAD tool applied to functional brain images for the diagnosis of Alzheimer's disease**. IEEE Transactions on medical imaging. 31 (2): 207–216. 2012.

70. EL-DAHSHAN, E.-S. A. et al. **Computer-aided diagnosis of human brain tumor through MRI:** a survey and a new algorithm. Expert systems with Applications. 41 (11): 5526–5545. 2014.

71. SINGH, N.; CHOUDHARY, N. **A Survey:** Brain tumor detection techniques of Computer aided diagnosis through MRI image. International Journal of Computer Science Issues (IJCSI). 12 (6): 148. 2015.

72. GHAFOORIAN, M. et al. **Automated detection of white matter hyperintensities of all sizes in cerebral small vessel disease**. Medical Physics. 43 (12): 6246–6258. 2016.

73. FILIPPI, M. et al. **Intra-and inter-observer agreement of brain MRI lesion volume measurements in multiple sclerosis**. Brain. 118 (6): 1593–1600. 1995.

74. GRIMAUD, J. et al. **Quantification of MRI lesion load in multiple sclerosis: a comparison of three computer-assisted techniques**. Magnetic resonance imaging. 14 (5): 495–505. 1996.

75. UDUPA, J. K. et al. **Multiple sclerosis lesion quantification using fuzzy connectedness principles**. IEEE Transactions on Medical Imaging. 16 (5): 598–609. 1997.

76. PARODI, R. C. et al. **Growing region segmentation software (GRES) for quantitative magnetic resonance imaging of multiple sclerosis: i**ntra-and inter-observer variability: a comparison with standard manual contouring method. European radiology. 12(4): 866–871. 2002.

77. ASHTON, E. A. et al. **Accuracy and reproducibility of manual and semiautomated quantification of MS lesions by MRI**. Journal of Magnetic Resonance Imaging. 17 (3): 300–308. 2003.

78. SHI, L. et al. **Automated quantification of white matter lesion in magnetic resonance imaging of patients with acute infarction**. Journal of neuroscience methods. 213 (1): 138–146. 2013.

79. SAAD, N. et al. **Automatic Brain Lesion Detection and Classification Based on Diffusion-Weighted Imaging using Adaptive Thresholding and a Rule-Based Classifier**. International Journal of Engineering and Technology (IJET). 6 (6). 2014.

80. WANG, R. et al. **Automatic segmentation of white matter lesions on magnetic resonance images of the brain by using an outlier detection strategy**. Magnetic resonance imaging. 32 (10): 1321–1329. 2014.

81. KADKHODAEI, M. et al. **Automatic segmentation of multimodal brain tumor images based on classification of super-voxels**. In: Engineering in Medicine and Biology Society (EMBC). 2016 IEEE 38th Annual International Conference of the. IEEE. 5945–5948. 2016.

82. SOLTANINEJAD, M. et al. **Automated brain tumor detection and segmentation using superpixel-based extremely randomized trees in FLAIR MRI**. International Journal of Computer Assisted Radiology and Surgery. 1–21. 2016.

83. ACHANTA, R. et al. **SLIC superpixels compared to state-of-the-art superpixel methods**. IEEE transactions on pattern analysis and machine intelligence. 34 (11): 2274–2282. 2012.

84. Images and Visual Representation Laboratory. **SLIC Superpixels**. http: //ivrl.epfl.ch/research/superpixels. Accessed in: 2018-01-04.

85. SCHMIDHUBER, J. **Deep learning in neural networks:** An overview, Neural networks. 61: 85–117. 2015.

86. NAJAFABADI, M. M. et al. **Deep learning applications and challenges in big data analytics**. Journal of Big Data. 2 (1): 1. 2015.

87. VAN GINNEKEN, B. et al. **Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans**. In: Biomedical Imaging (ISBI). 2015 IEEE 12th International Symposium on. IEEE. 286–289. 2015.

88. DA SILVA, G. L. et al. **Lung nodules diagnosis based on evolutionary convolutional neural network**. Multimedia Tools and Applications. 1–17. 2017.

89. LECUN, Y. et al. **Handwritten digit recognition with a back-propagation network**. In: Neural Information Processing Systems (NIPS). 1989.

90. KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. **Imagenet classification with deep convolutional neural networks**. In: Advances in neural information processing systems. 1097–1105. 2012.

91. LECUN, Y.; BENGIO, Y.; HINTON, G. **Deep learning**. Nature. 521 (7553): 436–444. 2015.

92. BENGIO, Y.I.; GOODFELLOW, J.; COURVILLE, A. **Deep Learning**. MIT Press, Cambridge, MA. http://www.deeplearningbook.org. 2016.

93. CIRE, D. et al. **Multi-column deep neural network for traffic sign classification**. Neural Networks. 32: 333–338. 2012.

94. NING, F. et al. **Toward automatic phenotyping of developing embryos from videos**. IEEE Transactions on Image Processing. 14 (9): 1360–1371. 2005.

95. SERMANET, P. et al. **Pedestrian detection with unsupervised multi-stage feature learning**. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 3626–3633. 2013.

96. TOMPSON, J. et al. **Efficient object localization using convolutional networks**. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 648–656. 2015.

97. GARCIA, C.; DELAKIS, M. **Convolutional face finder:** A neural architecture for fast and robust face detection. IEEE Transactions on pattern analysis and machine intelligence. 26 (11): 1408–1423. 2004.

98. TAIGMAN, Y. et al. **Deepface:** Closing the gap to human-level performance in face verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 1701–1708. 2014.

99. WANG, T. et al. **End-to-end text recognition with convolutional neural networks**. In: Pattern Recognition (ICPR), 2012 21st International Conference on. IEEE. 3304–3308. 2012.

100. LI, R. et al. **Deep learning based imaging data completion for improved brain disease diagnosis**. In: International Conference on Medical Image Computing and Computer Assisted Intervention. Springer. 305–312. 2014.

101. BROSCH, T. et al. **Modeling the variability in brain morphology and lesion distribution in multiple sclerosis by deep learning**. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham. 462–469. 2014.

102. AREVALO, J. et al. **Representation learning for mammography mass lesion classification with convolutional neural networks**. Computer methods and programs in biomedicine. 127: 248–257. 2016

103. ZHANG, W. et al. **Deep convolutional neural networks for multi-modality isointense infant brain image segmentation**. NeuroImage 108 214–224. 2015.

104. CARASS, A. et al. **Simple paradigm for extra-cerebral tissue removal:** algorithm and analysis, NeuroImage. 56 (4): 1982–1992. 2011.

105. ESKILDSEN, S. F. et al. **BEaST:** brain extraction based on nonlocal segmentation technique. NeuroImage. 59 (3): 2362–2373. 2012.

106. IGLESIAS, J. E. et al. **Robust brain extraction across datasets and comparison with publicly available methods**. IEEE transactions on medical imaging. 30 (9): 1617–1634. 2011.

107. SADANANTHAN, S. A. et al. **Skull stripping using graph cuts**. NeuroImage. 49 (1): 225–239. 2010.

108. SATO, Y.; GERIG, G. **MICCAI:** medical image computing and computer assisted intervention1. Academic Radiology. 10 (12): 1339–1340. 2003.

109. WANG, Y. et al. **Robust deformable-surface-based skull-stripping for large-scale studies**. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Berlin, Heidelberg. 635–642. 2011.

110. SMITH, S. M. **Fast robust automated brain extraction**. Human brain mapping. 17 (3): 143–155. 2002.

111. BAUER, S.; NOLTE, L. P.; REYES, M. **Skull-stripping for tumor-bearing brain images**. arXiv preprint arXiv:1204.0357.2012.

112. GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing.** 3. ed. Prentice Hall. 128. ISBN: 9780131687288. 2008

113. LECUN, Y. et al. **Gradient-based learning applied to document recognition**. Proceedings of the IEEE 86 (11). 2278–232. 1998.

114. MOORE, G. E. **Cramming more components onto integrated circuits**. Electronics. 38 (8): 114. 1965.

115. NATIONAL ELECTRICAL MANUFACTURES ASSOCIATION – NEMA. **Digital Imaging and Communications in Medicine (DICOM).** 3-1, 3, 5, 6 e 10. 2009.

116. THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern Recognition.** 4. ed. Elsevier. ISBN: 978-1-59749-272-0. 2009.

117. BROWN, L. G. **A survey of image registration techniques**. ACM Computing Surveys (CSUR). 24 (4): 325–376. 1992.

118. CRUM, W. R.; HARTKENS, T.; HILL, D. L. G. **Non-rigid image registration:** theory and practice. The British Journal of Radiology. 77: 140–153. 2004.

119. GOSHTASBY, A. A.. **2-D and 3-D image registration for medical, remote sensing, and industrial applications**. John Wiley & Sons. 2007.

120. HAFEMANN, L. G. **An analysis of deep neural networks for texture classification**. 2014.

121. BENGIO, Y. **Learning deep architectures for AI**. Foundations and trends in Machine Learning. 2(1): 1–127. 2009.

122. CIREŞAN, D. et al. **A committee of neural networks for traffic sign classification.** In: Neural Networks (IJCNN). The 2011 International Joint Conference on. IEEE. 1918-1921. 2011.

123. SRIVASTAVA, N. et al. **Dropout:** a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research, 15(1): 1929–1958. 2014.

124. MCCULLOCH, W. S.; PITTS, W. **A logical calculus of the ideas immanent in nervous activity**. The bulletin of mathematical biophysics. Springer 5(4): 115–133. 1943.

125. HORNIK, K.; STINCHCOMBE, M.; WHITE, H. **Multilayer feedforward networks are universal approximators**. Neural networks. Elsevier. 2(5): 359–66. 1989.

126. FULKERSON, B.; VEDALDI, A.; SOATTO S. **Class segmentation and object localization with superpixel neighborhoods**. in: International Conference on Computer Vision (ICCV). 2009.

127. YANG, Y. et al. **Layered Object Detection for Multi-Class Segmentation**, in: Computer Vision and Pattern Recognition (CVPR). 2010.

128. GOULD, S. et al. **Multi-class segmentation with relative location prior**. International Journal of Computer Vision (IJCV). 80(3): 300–316. 2008.

129. ZITNICK, C. L.; KANG, S. B. **Stereo for image-based rendering using image over-segmentation.** International Journal of Computer Vision (IJCV), 75(1): 49–65. 2007.

130. LI, Y. et al. **Lazy snapping**. ACM Transactions on Graphics (SIGGRAPH), 23(3): 303–308. 2004.

131. MORI, G. **Guiding model search using segmentation.** in: IEEE International Conference on Computer Vision (ICCV). 2005.

132. WXWIDGETS. **wxWidgets Cross-plataform GUI Library**. https://www.wxwidgets.org/. Accessed in: 2018-04-08.

133. INSIGHT SEGMENTATION AND REGISTRATION TOOLKIT, **ITK**, http://www.itk.org/. Accessed in: 2018-04-08.

134. OPENCV. **OpenCV [Open Source Computer Vision]**. http://opencv.org/. Accessed in: 2018-04-08.

135. VISUALIZATION TOOLKIT. **VTK**. http://www.vtk.org/. Accessed in: 2018-04-08.

136. KERAS. **Keras**. http://keras.io/. Accessed in: 2018-04-08.

137. THEANO. **Theano**. http://deeplearning.net/software/theano/. Accessed in: 2018-04-08.

138. TENSORFLOW. **Tensorflow**. https://www.tensorflow.org/. Accessed in: 2018-04-08.

139. İLSEVER, M.; ÜNSALAN , C. **Pixel-based change detection methods**. In: Two-Dimensional Change Detection Methods. Springer, London. 7–21. 2012.

140. NETTO, S. M. B. et al. **Voxel-based comparative analysis of lung lesions in CT for therapeutic purposes**. Medical & biological engineering & computing. 55(2): 295–314. 2017.

141. DASA. **DASA.** https://www.dasa.com.br/. Accessed in: 2018-06-05.

A. WML detection software

This appendix presents the prototype of the WML Detection Software, developed during the period of elaboration of the thesis to support the research and, at the same time, provide the necessary resources for specialists for MRI visualization and WML segmentation. Some of these resources are: visualization of the slices, zoom, pan, rotation, contrast window adjustment, manual and automatic segmentation and save markings in file.

This software was developed for WML marking. The main objective is to deliver the necessary tools for radiologists to create a database of WML markings. Therefore, there was a concern for the graphical user interface (GUI) to be friendly. Moreover, the faster the radiologist learns to handle the system, the faster the volumes are marked.

Although this software has specific initial use for brain MRI volumes, it can be used to visualize any image using the DICOM standard and can easily be adapted to images of other fields of study.

The automatic segmentation of WML from the software was developed following the method of Chapter 3 which uses only FLAIR image and is faster than other methods described in this thesis.

This appendix is structured as follows. The Section A.1. describes the requirements raised for software specification. Section A.2. describes the use cases defined from the raised requirements. Section A.3. shows the modeling and developmental details used to develop software that addresses the use cases. Finally, Section A.4. shows the software GUI and how to perform its features.

A.1. Requirements and Specifications

This section shows the gathered requirements and specifications for the development of the MRI Visualization and WML segmentation Software. They are specified below:

1. Read and load DICOM volumes. The user can open a volume of DICOM images by selecting the folder and loading it to the program.

Priority	(X) Essential	( ) Important	( ) Desirable
Effort	( ) High	(X) Medium	( ) Low

2. Visualization of DICOM Volume. The user can view the DICOM volume slices in the program.

Priority	(X) Essential	( ) Important	( ) Desirable
Effort	( ) High	(X) Medium	( ) Low

3. Interaction with volume. The user can zoom, pan, rotate, adjust the contrast window, and also obtain pixel information from the volume sections using mouse commands in the program.

Priority	( ) Essential	(X) Important	( ) Desirable
Effort	( ) High	(X) Medium	( ) Low

4. Manipulate WML markings manually. The user can manually add, edit and remove WML markings over the volume slices using the mouse.

Priority	(X) Essential	( ) Important	( ) Desirable
Effort	( ) High	(X) Medium	( ) Low

5. Perform WML automatic segmentation. The user can automatically segment WMLs in the program. The segmented markings should be displayed after the process.

Priority	( ) Essential	( ) Important	(X) Desirable
Effort	(X) High	( ) Medium	( ) Low

6. WML markings exhibition. The user can show or hide WML markings displayed on the volume section in the program.

Priority	( ) Essential	( ) Important	(X) Desirable
Effort	( ) High	( ) Medium	(X) Low

7. Maintain a WML markings history. The user can save the WML markings description on a file in the program. This file can be loaded back later. The markings description loaded from file should be displayed on viewer so the user can continue his work.

Priority	(X) Essential	( ) Important	( ) Desirable
Effort	( ) High	(X) Medium	( ) Low

A.2. Use Cases

This section shows the Use Cases Diagram and describes the Use Cases that have been defined to meet the requirements gathered in section A.1. Figure 31 shows this diagram. The Use Cases are described after.

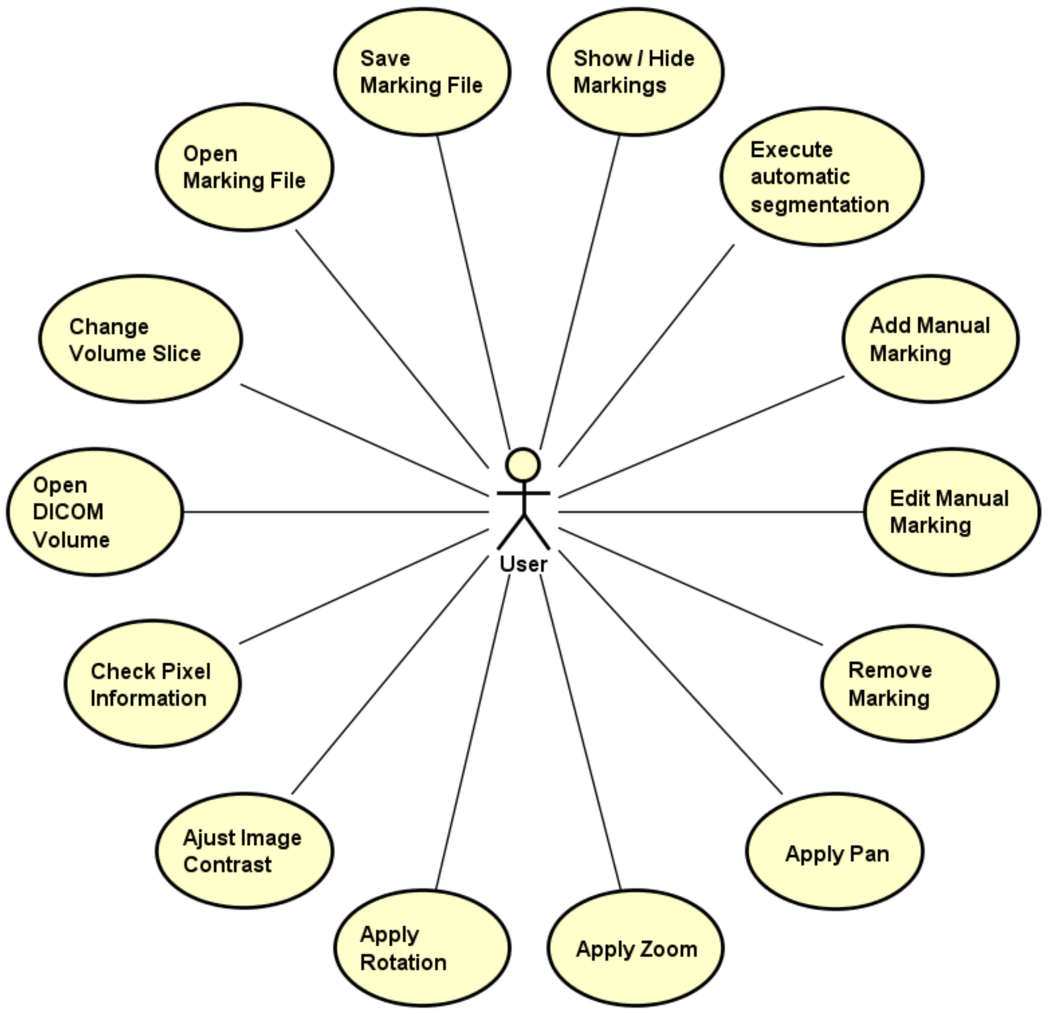


Figure 31: Use Cases Diagram of the WML segmentation software.

1. Open DICOM Volume.

Pre-conditions	None.
Description	The user searches for a folder with DICOM files in a window. After choosing the folder, the program will read and load your information and use them to generate a seismic image in the GUI volume viewer.

2. Change Volume Slice.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user clicks on tool to move to next slice or previous slice. The viewer exhibited the desired slice after clicking the tool.

3. Open Marking File.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user looks for a tag file in a window. After choosing the file, the markings described in the file are displayed in the viewer on the slice.

4. Save Marking File.

Pre-conditions	Volume slice displayed in the viewer. At least one marking must exist.
Description	The user chooses a folder to save a file containing the description of markings made in the program.

5. Show / Hide Markings.

Pre-conditions	Volume slice displayed in the viewer. At least one marking must exist.
Description	The user clicks on the marking exhibition tool. The program viewer shows / hides the markings.

6. Execute automatic segmentation.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user clicks on the WML auto-targeting tool. The program performs segmentation and draws up the resulting markup.

7. Add Manual Marking.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user clicks on the manual marking tool. Then click on the viewer repeatedly to draw the marking.

8. Edit Manual Marking.

Pre-conditions	Volume slice displayed in the viewer. Manual marking added.
Description	The user clicks and presses on the points of the manual segmentation and moves them to edit its shape.

9. Remove marking.

Pre-conditions	Volume slice displayed in the viewer. At least one marking must exist.
Description	The user clicks the marking removal tool and then clicks on the markings that he wants to remove. The program removes the chosen markings from the viewer.

10. Apply Zoom.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user rotates the mouse wheel over the volume section to zoom in or zoom out.

11. Apply Pan.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user clicks and holds the mouse wheel over the volume section and moves the mouse to apply pan.

12. Apply Rotation.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user presses the Ctrl button and moves the mouse cursor to apply the rotation.

13. Adjust Contrast Window.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user clicks and holds the left mouse button on the seismic image and moves the mouse to apply the contrast adjustment.

14. Check Pixel Information.

Pre-conditions	Volume slice displayed in the viewer.
Description	The user moves the mouse cursor to the pixel that he wants to see the location and value.

A.3. Development

This section contains information on the development phase of the software.

A.3.1. Platforms and Technologies

The program was written using the C ++ 14 and Python 2.7 languages.

The C ++ code was written in the Microsoft Visual Studio 2017 IDE and using the Visual C ++ Compiler, but because it uses the C ++ 14 standard and multiplatform libraries, it can be recompiled to other compilers such as MinGW (Windows), GCC (Linux), and CLang MacOS). For the development of the GUI, the wxWidgets library [132] was used. ITK [133] and OpenCV libraries [134] were used for supporting the image processing algorithms on MRI volumes. For visualization and interaction with the volume slices and markings, the VTK library [135] was used.

The Python code was also written in the Microsoft Visual Studio 2017 IDE with support of the Anaconda package manager. The Python interpreter and used packages are also cross-platform. The Python code is completely used to machine learning part of WML segmentation (from patches generation to classification). In addition to the base packages, the main packages used are Keras [136], Theano [137] and Tensorflow [138], all of which are used to classify candidates through CNN.

Table 1: Summary of platforms and technologies used in software development.

Table 10: Summary of platforms and technologies used to develop the software.

Programming languages	C++ 14, Python 2.7
IDE used for development	Visual Studio 2017
Supported compilers / interpreters	Visual C++ Compiler, Python 2.7 Interpreter
Main Libraries / packages	wxWidgets 3.0.2, ITK 4.6.0, VTK 6.1.0, OpenCV 2.4.9, Keras 2.1.3, Theano 1.0.1, Tensorflow 1.2.1
Platforms	Windows (portable to MacOS and Linux)

A.3.2. Modeling and Architecture

Following, it is shown the modeling and architecture for software development.

The architecture of the software uses the Model-View-Presenter (MVP) standard, which is a variation of the Model-View-Controller architecture standard and is more aimed towards building GUI. In this standard, classes are divided into three layers: Model, View, and Presenter. The Model layer accommodates the application model / data and provides behaviors to access it systematically. It should be independent, should have no association with GUI issues. Thus, it is possible to reuse the same model for different GUI types. The View layer accommodates the interface to show the model data and to direct the user event results from it to the Presenter layer. The View layer knows the Model, but the opposite is not true. Finally, the Presenter layer acts between View and Model. It receives the data from the Model layer and treats it to display in the View and receives the event-generated entries from the View to update the Model. Figure 32 illustrates the MVP architecture standard.

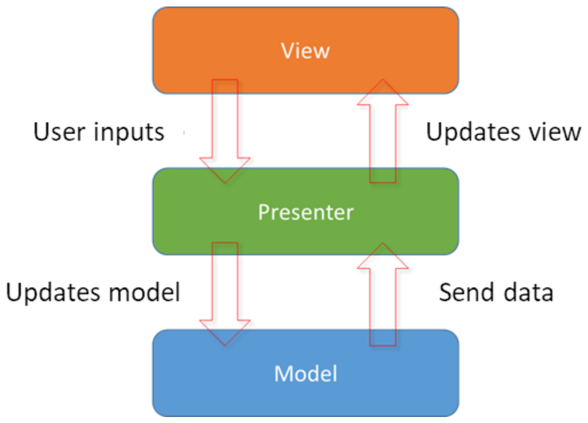


Figure 32: Model-View-Presenter standard.

Figure 33 shows modeling of the software through a Class Diagram. The attributes and methods were excluded for better visualization. The blue classes represent the Model layer, the orange classes represent View layer and the green classes represent Presenter layer.

Starting with the Model Layer classes, the BrainVolumeReader class is responsible for reading a folder with DICOM files. The output of this class is an object of the BrainVolume class, which is the representation of the brain MRI volume whose information was in the DICOM files.

The MarkingsReader class is responsible for read files containing markings descriptions. Once it reads a file, it instantiates a MarkingsMap object, which manages markings descriptions by adding, editing or removing them. MarkingsWriter class, on the other hand, is responsible for save markings descriptions from a MarkingsMap in a file on disk.

The BrainLesionSegmenter is the class responsible for the automatic WML segmentation on a BrainVolume. To do this, it executes these substeps: brain segmentation (performed by BrainSegmenter), histogram matching (done by HistogramMatching class), superpixel segmentation (performed by

SLIC0Segmenter), and superpixel classification (performed by CNNClassificer that uses internally the CNNPatchesGenerator to generate patches from superpixels). The result of the execution of the BrainLesionSegmenter procedures is a MarkingsMap object.

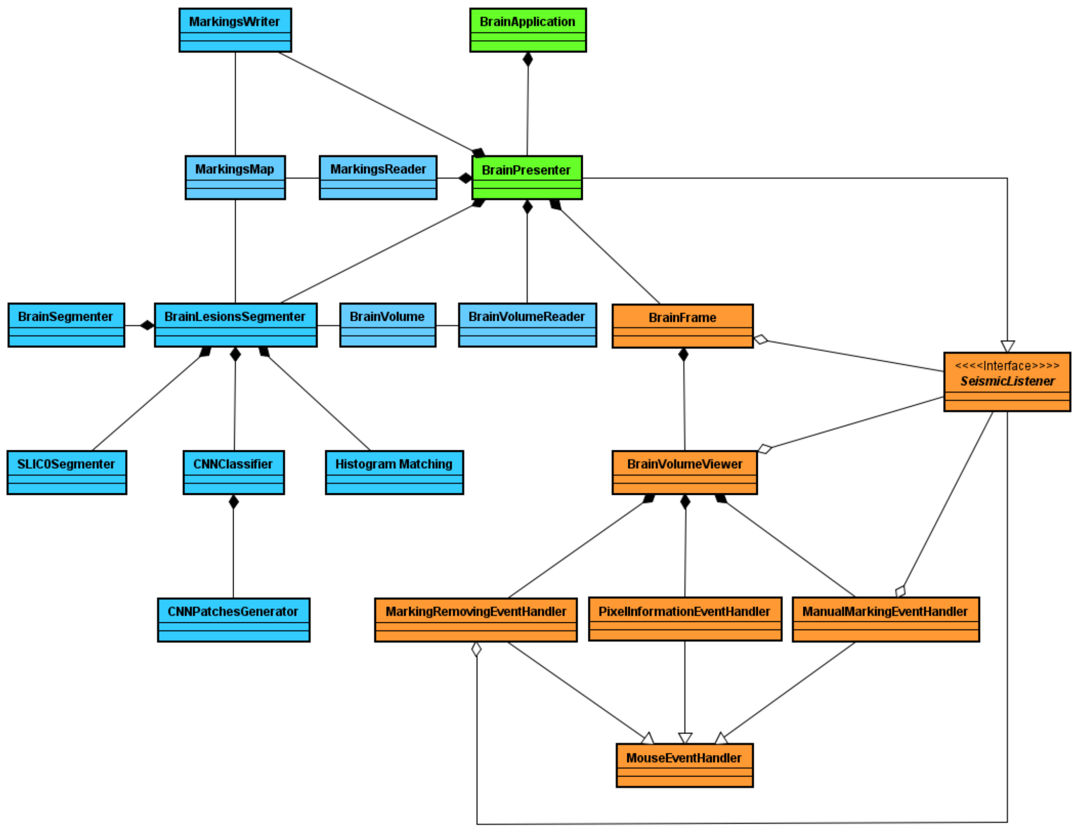


Figure 33: Class Diagram.

Most of the Model Layer classes were implemented using ITK and OpenCV, except the CNNPatchesGenerator and CNNClassifier classes that mainly used the Keras and Theano / Tensorflow packages.

In the View layer, the BrainFrame class is responsible for building the GUI, implemented with the support of wxWidgets. It inherits from wxFrame, which wxWidgets window class. It features the menu bar, tool bar and a BrainVolumeViewer, which is the class that uses the VTK library to display the graphical representation of the MRI volume and graphical representation of markings. For handling events that originate from user interaction with the viewer, the PixelInformationEventHandler, ManualMarkingEventHandler and MarkingRemoverEventHandler classes are used. The first one is responsible for getting the coordinates of the mouse cursor over the image and displaying them in the viewer along with the value of the pixel whose cursor is over and the number of current slice. The second one, when activated, is responsible for capturing the left-click coordinates on the image to draw markings. The third one, when activated, is responsible for capturing the left-click coordinates on the image to remove markings. These handlers classes inherit from MouseEventHandler, which provides the "shell" of the mouse event handling, such as a method that converts the coordinates of the mouse cursor to image coordinates. In turn, it inherits from vtkCommand, which is the VTK event handling class. Finally, there is the BrainListener class. It is nothing more than a communication interface with BrainPresenter, informing it about the interactions made on the GUI.

In the Presenter layer, we have two classes. The first is BrainPresenter, which receives window interactions through the BrainListener interface and uses them to update the objects of the Model layer and then updates the View layer with the changes. The second class is SeismicApplication, which is only responsible for instantiating and destroying SeismicPresenter. It works like a main function one of a normal software and inherits of class wxApp, obligatory for initialization of wxWidgets GUI.

To better understand the two of the main functions of the program, two sequence diagrams were created (Figures 34 and 35). The classes displayed in these diagrams show the colors of the MVP layer that they are included.

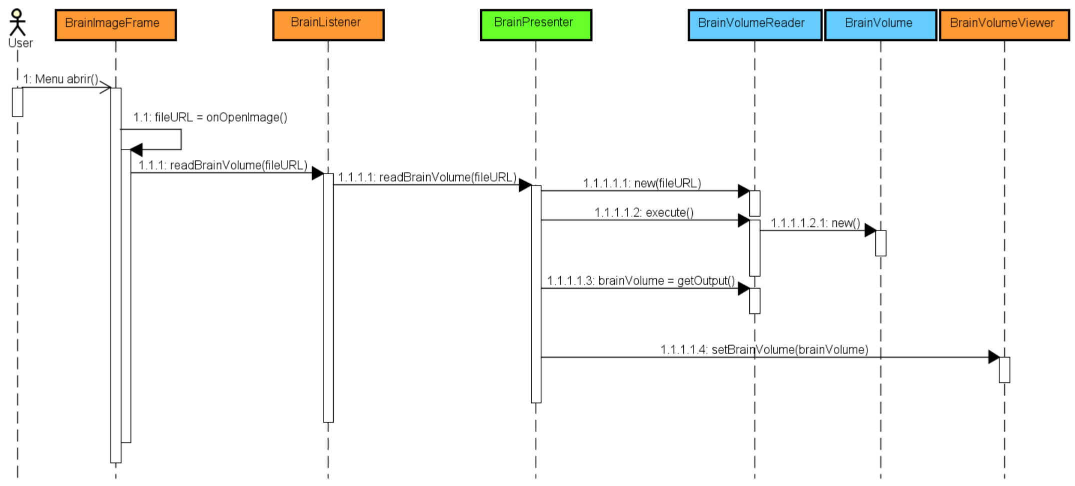


Figure 34: Sequence Diagram of DICOM volume reading.

The first diagram shows the relationship of the classes in the DICOM volume reading and loading tasks. First the user clicks on the submenu or tool "Abrir volume ..." contained in the "Arquivo" menu. This triggers the event that activates the BrainFrame method onOpenImage method which in turn displays a directory selection window. Once the directory is chosen, BrainFrame notifies BrainPresenter through the BrainListener interface, which in turn instantiates a BrainVolumeReader so that it reads the file. If the file is valid, BrainVolumeReader instantiates a BrainVolume with the file information. The BrainFrame once again calls the BrainPresenter through the BrainListener interface to obtain the volume information and passes it to the BrainVolumeViewer which creates a graphical representation of the volume and displays it.

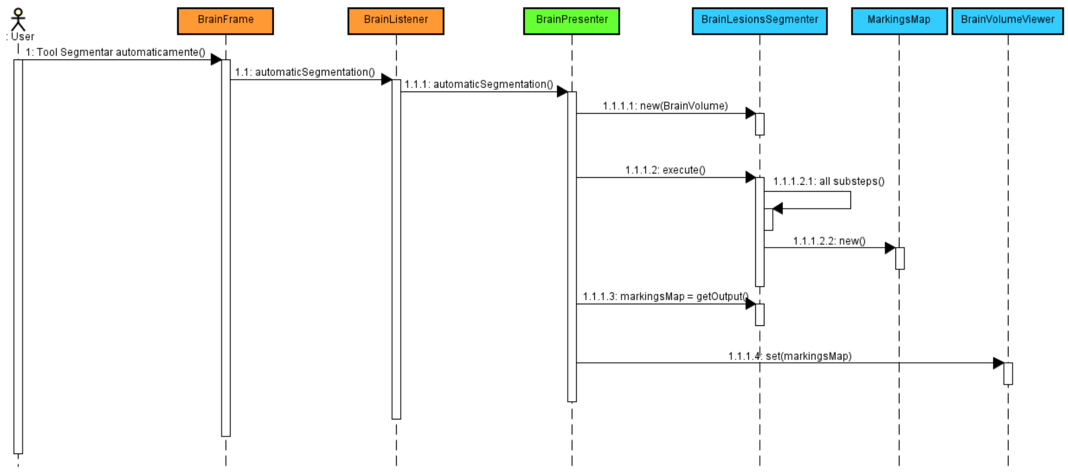

: User
BrainFrame
BrainListener
BrainPresenter
BrainLesionsSegmenter
MarkingsMap
BrainVolumeViewer
1: Tool Segmentar automaticamente()
1.1: automaticSegmentation()
1.1.1: automaticSegmentation()
1.1.1.1: new(BrainVolume)
1.1.1.2: execute()
1.1.1.2.1: all substeps()
1.1.1.2.2: new()
1.1.1.3: markingsMap = getOutput()
1.1.1.4: set(markingsMap)

Figure 35: Sequence Diagram of automatic WML segmentation.

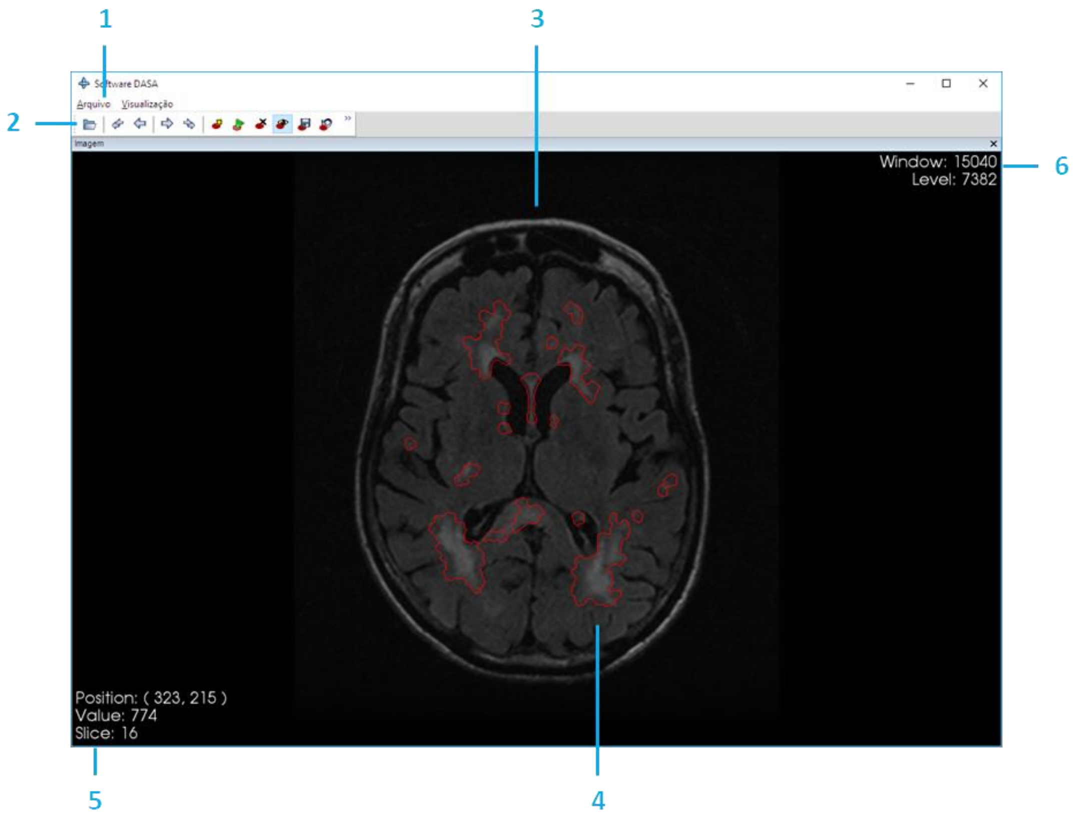
The second sequence diagram shows the class relationships in the WML auto-segmentation task. To perform this task, the user should click on the "Segmentar automaticamente" tool. BrainFrame will tell BrainPresenter about the command through BrainListener. BrainPresenter will instantiate a BrainLesionsSegmenter by passing the currently used BrainVolume. The BrainLesionsSegmenter will execute all targeting sub-steps involving the BrainSegmenter, HistogramMatching, SLIC0Segmenter, and CNNClassifier classes (which internally uses CNNPatchesGenerator). Finally, a MarkingsMap resulting from this operation is returned to BrainPresenter and BrainPresenter sends the markup information to the BrainVolumeViewer. BrainVolumeViwer will create graphical representations of the markings and display them.

A.4. Software

This section introduces the software of WML detection software. It is shown and explained its GUI as well as how to use its features.

A.4.1. Graphical User Interface

Figure 36 shows the typical GUI of the software. The interface components have been numbered and are defined below.



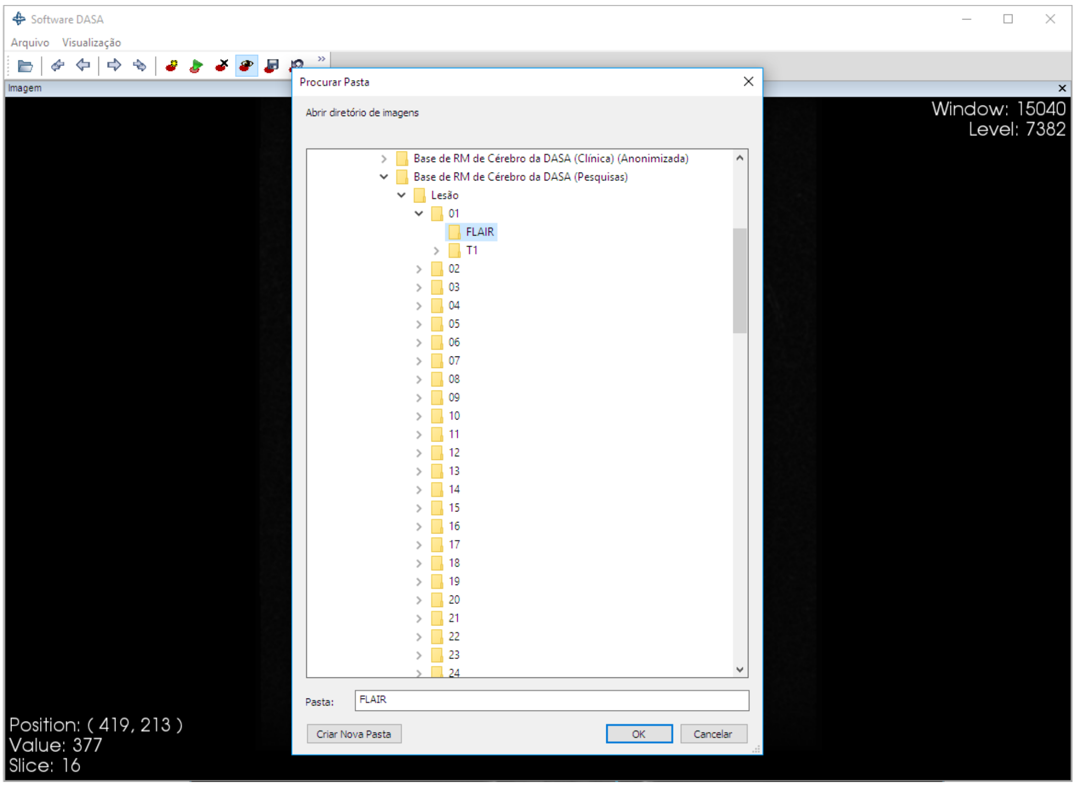
Window: 15040
Level: 7382
Position: (323, 215)
Value: 774
Slice: 16

Figure 36: GUI Components.

1. Menubar. In it, the user chooses the features he wants to run in the software.
2. Toolbar. As in the menubar, here the user chooses the functionality that he wants to perform.
3. Volume section. Current slice of the volume loaded in the program.
4. WML marking. One of the examples of markings made on the slice.
5. Pixel information. These values are displayed when the user moves the mouse cursor over the image. They are the coordinates of the image and the pixel value of that coordinate. If the cursor is outside the image, the message "Fora da imagem" is displayed.
6. Contrast window information. They are information about the volume contrast window and can be adjusted to improve viewing of specific points.

A.4.2. Features and How to Use

For the user to open a DICOM volume, he must first select the option in the menu or toolbar. After that, the directory selection window will be displayed, as in Figure 37.



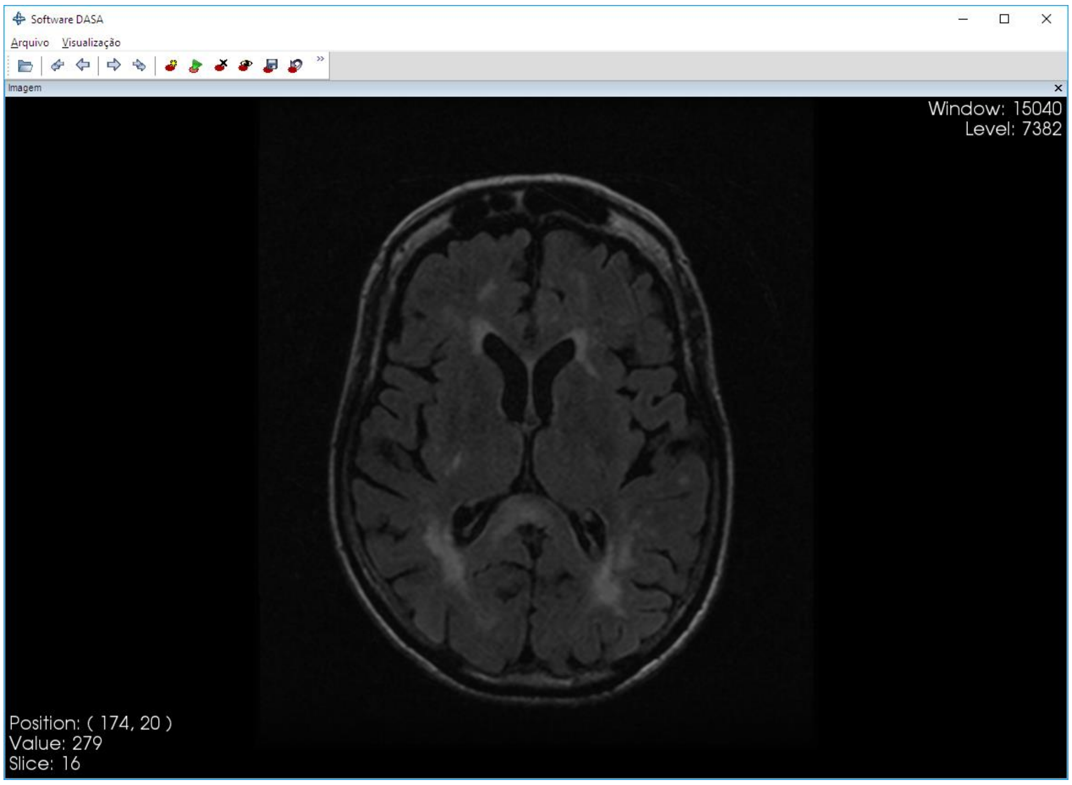
Pasta: FLAIR

Window: 15040
Level: 7382

Position: (419, 213)
Value: 377
Slice: 16

Figure 37: DICOM directory selection window.

Once the directory is chosen and "Abrir" button is clicked, the image is displayed in the main window viewer, as in Figure 38.

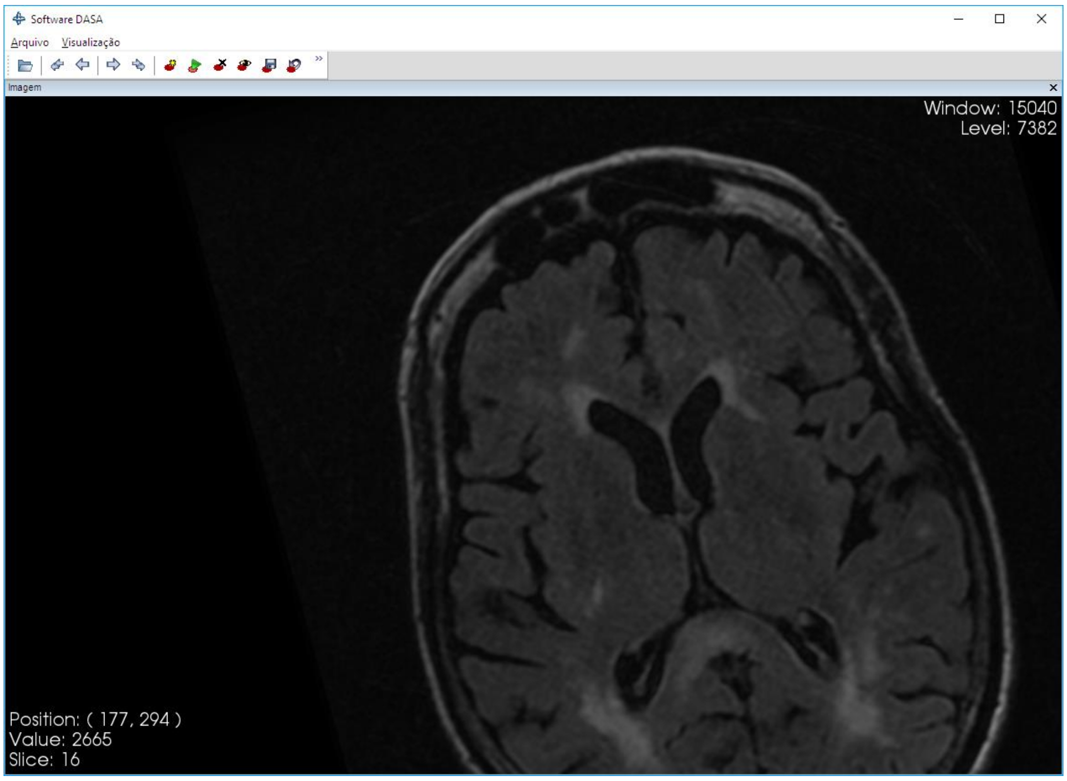


Window: 15040
Level: 7382

Position: (174, 20)
Value: 279
Slice: 16

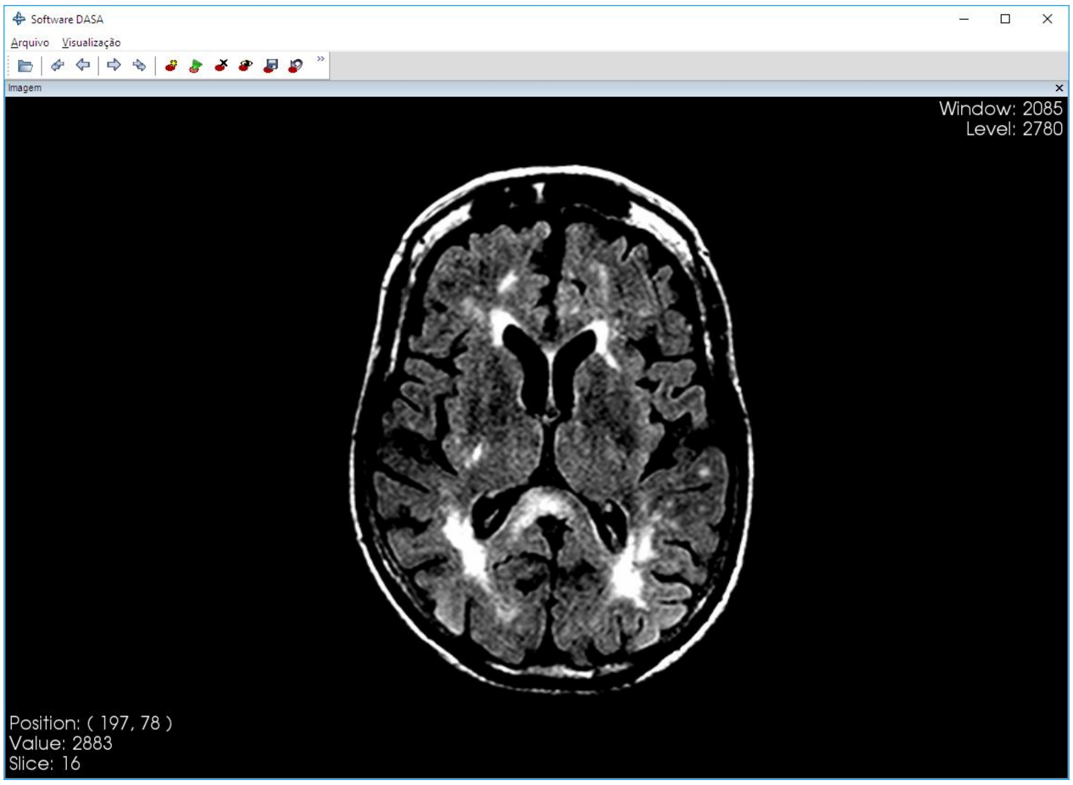
Figure 38: Displaying volume slice.

Once the volume slice is displayed it is possible to adjust display parameters, such as zoom and pan (Figure 39) and contrast window (Figure 40).



Window: 15040
Level: 7382
Position: (177, 294)
Value: 2665
Slice: 16

Figure 39: Applying zoom and pan on volume slice.



Window: 2085
Level: 2780
Position: (197, 78)
Value: 2883
Slice: 16

Figure 40: Contrast Window Adjustment.

To zoom in, simply rotate the mouse wheel forward on the volume slice, and to zoom out, rotate the mouse wheel back. To apply the pan, click and hold the mouse wheel and move the mouse to where you want to move the image.

To adjust the contrast, click on the image with the left mouse button and move it.

When you move your mouse cursor over the image, you will see information about the pixel it is on. This is displayed in the lower left corner of the viewer.

To manually segment, user must click on the "Segmentar manualmente" tool. After that, click on the points around the region he wants to mark. The dots are connected by splines curves. Figure 41 shows this.

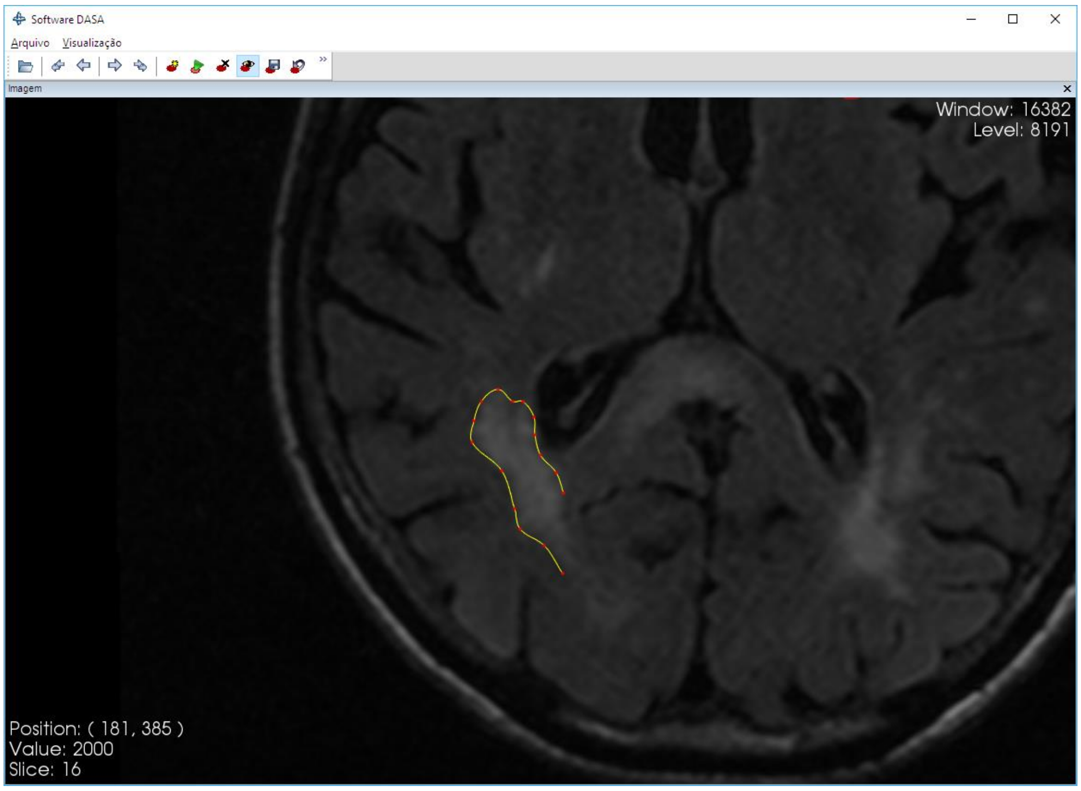


Figure 41: Manual segmentation process.

Once you've completed marking, you can adjust it. To do this, click on one of the marking points and drag to adjust the contour. This is shown in Figure 42.

Other features include:

- Remove the markings by clicking on the "Remover marcações" tool and then click on the markings to remove.
- Automatically segment by clicking the "Segmentar automaticamente" tool.
- Show and hide the markings by clicking on the "Mostrar marcações" tool.

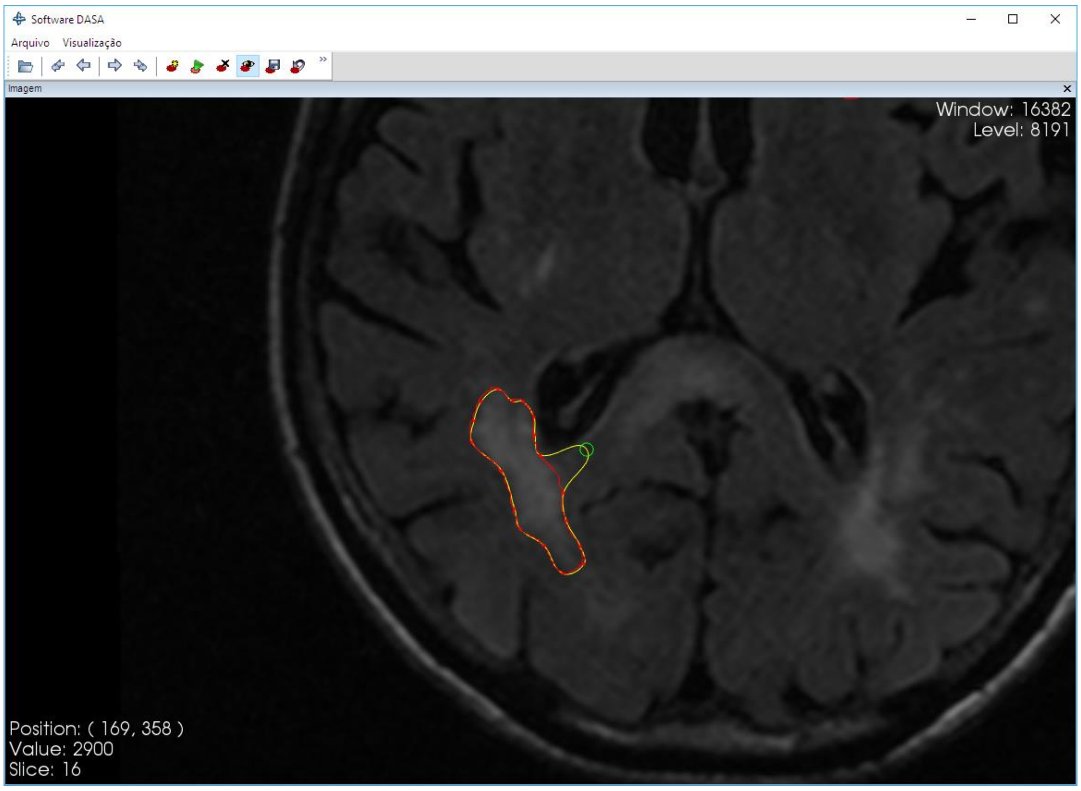


Figure 42: Manual segmentation process.