Alma Mater Studiorum · University of Bologna

**School of Science**
**Department of Physics and Astronomy**
**Master Degree in Physics**

# Implementation of an Automated Pipeline for the Identification of Ground Glass Opacities on CT Scans of Patients Affected by COVID-19

**Supervisor:**
**Prof. Gastone Castellani**

**Co-supervisor:**
**Dr.Nico Curti**

**Submitted by:**
**Riccardo Biondi**

Academic Year 2019/2020

# Abstract

COronaVirus Disease (COVID-19) has widely spread all over the world since the beginning of 2020. It is acute, highly contagious, viral infection mainly involving the respiratory system.

Chest CT scans of patients affected by this condition have shown peculiar patterns of Ground Glass Opacities (GGO) and Consolidation (CS) related to the severity and the stage of the disease.

In this scenario, the correct and fast identification of these patterns is a fundamental task. Up to now this task is performed mainly using manual or semi-automatic techniques, which are time-consuming (hours or days) and subjected to the operator expertise.

In this work of thesis, I have developed and implemented an automated pipeline for the identification of GGO and CS patterns in chest CT scans. To achieve this purpose, I have applied colour quantization and obtained a hard classification based on voxel intensities. I have used the way in which digital images encodes colour to takes into account also properties related to the voxel neighbourhood, useful since lesions involve many close voxels.

The pipeline was tested on three different datasets and compared with manual annotation by checking specificity and sensitivity and with a blind evaluation made by experts. The results of these preliminary tests show that the pipeline is able to achieve segmentation with high specificity in less than $3\,min$. Even if the total lesion volume is underestimated, the pipeline has shown to achieve segmentation consistent with annotation.

# Contents

# Introduction

Since the end of 2019, COVID-19 has widely spread all over the world. Up to now, the gold standards for the diagnosis of this disease are the reverse transcription-polymerase chain reaction (RT-PCR) and the gene sequencing of sputum, throat swab and lower respiratory tract secretion [1].

An initial prospective made by Huang et al. [2] on chest CT scans of patients affected by COVID-19, has shown that the 98% of examined subjects have a bilateral patchy shadow or ground-glass opacity (GGO) and consolidation(CS). Their severity, shape and involved percentage of lungs were related to the stage of the disease [3]. In the end, other studies have monitored the change in volume and shape of these features on healed patients [4] to check their actual recovery. This behaviour can be seen in Figure 1, in which chest CT scans of patients with different severity of disease are compared.

**Figure 1:** *Groud Glass Opacity and Consolidation on chest CT scans of COVID-19 affected patients with different severity of the disease. From left to right we can observe an increment of the GGO areas in the lung.*

GGO and CS are not exclusive of COVID-19, but may be also caused by pulmonary edema, bacterial infection, other viral infection or alveolar haemorrage [5]. The combination of CT scan information and other diagnostic techniques like the RT-PCR mentioned above may help the diagnosis, the monitoring of the course of the disease and the checking of the recovery in healed patients. The study of these patterns may help to understand the infection pathogenesis, which is not well known since COVID-19 is a new illness.

Identification and quantification of these lesions in chest CT scans are a fundamental task. Up to now, the segmentation is made in a manual or in a semi-automatic way, which are time-consuming (several hours or days) and subjected to the operator experience, since requires the interaction with trained personnel. Moreover, these kinds of segmentation cannot be reproduced. To overcome these issues, an automatic and fast way for the segmentation is required.

This work of thesis, made in collaboration with the Department of Diagnostic and Preventive Medicine of the Policlinico Sant'Orsola - Malpighi, aims to develop an automatic pipeline for the identification of GGO and CS in chest CT of patients affected by COVID-19. The work was based and tested on chest CT scans provided by Sant'Orsola, but also public repositories [6] [7] were used as a benchmark.

We start our discussion by understanding what is a digital medical image, its physical meaning and digital representation, focusing mainly on Computed Tomography. After that a brief review on the main image segmentation techniques will be presented, focusing on the ones used for the implementation of the pipeline.

The discussion will continue by describing the main pipeline characteristics and its structure: We will see how colour quantization was used to achieve the segmentation and how the digital image properties were used to take into account different image features. We also discuss how a preliminary lung segmentation will help the identification performances. After that, we will continue the discussion by describing in details the pipeline implementation.

In the end, we will discuss the segmentation results. The pipeline performances were checked trough different method, like visual comparison with other segmentation techniques, quantitative comparison against manual annotation and blind evaluation by experts.

# Chapter 1

# Image Segmentation techniques

Image segmentation consists of the partitioning of an image into non-overlapping consistent regions that are homogeneous respect to some characteristics, such as intensity or texture [8]. The results of segmentation can be used to perform feature extraction, that provides fundamental information about organs or lesion volumes, cell counting, etc. If a patient performs many analysis during the time, image segmentation is a useful tool to monitor the evolution of particular lesions or tumours during therapy. Nowadays different non-invasive medical imaging techniques are available, such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI) or X-Ray imaging, that provide a map of the subject anatomy or function.

Image segmentation plays a crucial role in many medical imaging applications by automating or facilitate the delineation of anatomical structures and other regions of interest [8]. Manual segmentation is possible, but it is time-consuming and subject to operator variability making the results difficult to reproduce [9]. Therefore automatic or semi-automatic methods are preferable.

The major difficulty in medical image segmentation is the high variability in medical images. First and foremost, human anatomy itself shows major modes of variation [10]. Furthermore, the many different modalities (X-ray, CT, MRI, etc.) used to create medical images change the particular meaning of the data.

In this chapter I will provide a brief introduction about medical images, focusing mainly on Computed Tomography, which is the technique used to acquire the images segmented in this work. This part is followed by a discussion on the main image segmentation techniques.

## 1.1 Medical Images

A medical image is the representation of the internal structure or function of the anatomic region in the form of an array of picture elements (pixels/voxels). This discrete representation result from a process that maps each numerical value in a position of the space. The number of pixels used for this representation is an expression of the details with which the anatomy of function can be depicted. The physical meaning of the data changes according to the image acquisition modality (CT, MRI, PET, etc.); for instance, we can consider the computed tomography (CT) case:

Computed Tomography (CT) is a medical imaging technique which aims to reproduce cross-section images and the 3D anatomy of the examined subject. Each

data represents the capability of the corresponding volume to attenuate an x-ray beam. To match results from different scans, the beam attenuation is measured in Hounsfield Units(HU) :

$$CT - number = 1000 \times \frac{\mu - \mu_{H_2O}}{\mu_{H_2O}} \tag{1.1}$$

Where $\mu$ is the linear attenuation coefficient of the tissue, $\mu_{H_20}$ is the linear attenuation coefficient of the water, took as a reference. The linear attenuation coefficient of the air is considered as 0, so the corresponding CT number is $-1000$. For the bones, that have a density double than water, the CT number is 1000.

Medical images can be characterized by 5 properties: *pixel depth, photometric, interpretation, metadata and pixel data.*

**Pixel depth** Pixel Depth is the number of bits used to encode the information of each pixel. Each value of the tensor is an integer of floating-point number belonging to a domain related to the image format. It is related to the memory space necessary to store the image and the amount of information we want to store in each pixel. Higher bit allows us to store more information but requires more memory [11]. The most common are $[0, 255]$ for $8-$bit integer image, of $[0, 1]$ for floats. Also, other formats are available, like $16-$bit integers, widely used to represent medical images.

**Photometric Interpetation** The photometric interpretation specifies how the pixel data should be interpreted for the correct image display as a monochrome or colour image. We have to introduce the concept of *number of channels.* Monochrome images have only one sample per pixel (GL intensity). To encode colour information into pixels, we typically need multiple samples per pixel and to adopt a colour model that specifies how to obtain colours combining the samples. Colour may be used to encode blood flow direction and velocity in Doppler ultrasound [11]. In this works, I have used colour to consider also voxels neighbouring information and different image gamma

**Metadata** Metadata are the information that describes the image. Metadata is typically stored at the beginning of the file and contains (at least) the image matrix dimension and photometric interpretation. In medical image formats, it contains also information about how the image was produced or about the patients [11].

**Pixel Data** Numerical values of the pixels that are stored according to data type.

### 1.1.1 Medical Image Formats

Image file formats provide a standard way to store the information describing an image in a computer file. Moreover, file format describes how the image data are organized inside the image file and how the pixel should be interpreted by software for the correct loading and visualization.

Medical file formats can be divided into two categories: one that tries to standardize the images generated by diagnostic modality(e.g. DICOM), the other that try to facilitate the post-processing analysis (e.g. Nifti). Both of them store image data and metadata at the beginning of the file. [11].

**DICOM** , acronyms for Digital Imaging and COmmunications in Medicine, is not only a file format but also a network communication protocol. The added value of its adoption in terms of access, exchange, and usability of diagnostic medical images is, in general, huge. Dicom File format establishes that the pixels data cannot be separated from the description of the medical procedure which leads to the formation of the image itself. The header also contains patient information such as name, gender, age, etc. So the header allows the image to be self-descriptive. DICOM is born for only 2D images, so a 3D volume is described by a series of files containing the single slices. [11].

**Nifti** primary goal is to provide coordinated and targeted service, training, and research to speed the development and enhance the utility of informatics tools related to neuroimaging. This file format uses the header to store information about image orientation image centre and origin. This avoids left-right brain hemisphere ambiguity. Even if this file is born for neuroimaging, can be used also to store other kinds of images like chest CT. The format is supported by many viewers and image analysis software like 3D Slicer, ImageJ and OsiriX [11].

## 1.2 Review on Image Segmentation Methods

During the years, several segmentation methods have been developed and they are based on a lot of different approaches. These methods can be classified into several ways, for example, we can divide them into *supervised* or *unsupervised* if they require or not a set of training data, or can be classified according to the information type they use, like *Pixel classification methods*, which use only information about pixel intensity, or *Boundary following* methods, which use edge information, etc. In this section I will provide a brief review on the main segmentation methods, organized in the same way as in [8], that divides the methods into 8 categories:

1. Thresholding,
2. Region growing,
3. Classifiers,
4. Clustering,
5. Markov Random Fields models,
6. Artificial Neural Networks,
7. Deformable Models,
8. Atlas guided approaches.

### 1.2.1 Thresholding

Thresholding approach is very simple and it basically segments a scalar image by creating a binary partitioning of image intensities [8]. It can be applied on an image to distinguish regions with contrasting intensities and thus differentiate between

tissue regions represented within the image [9]. Figure 1.1 shows a histogram of a scalar image with two classes, the threshold-based approach attempts to determine an intensity value, called *threshold* which splits the desired classes [8]. To achieve the segmentation we can group all the pixels with intensity higher than the threshold in one class an all the remaining ones into other class

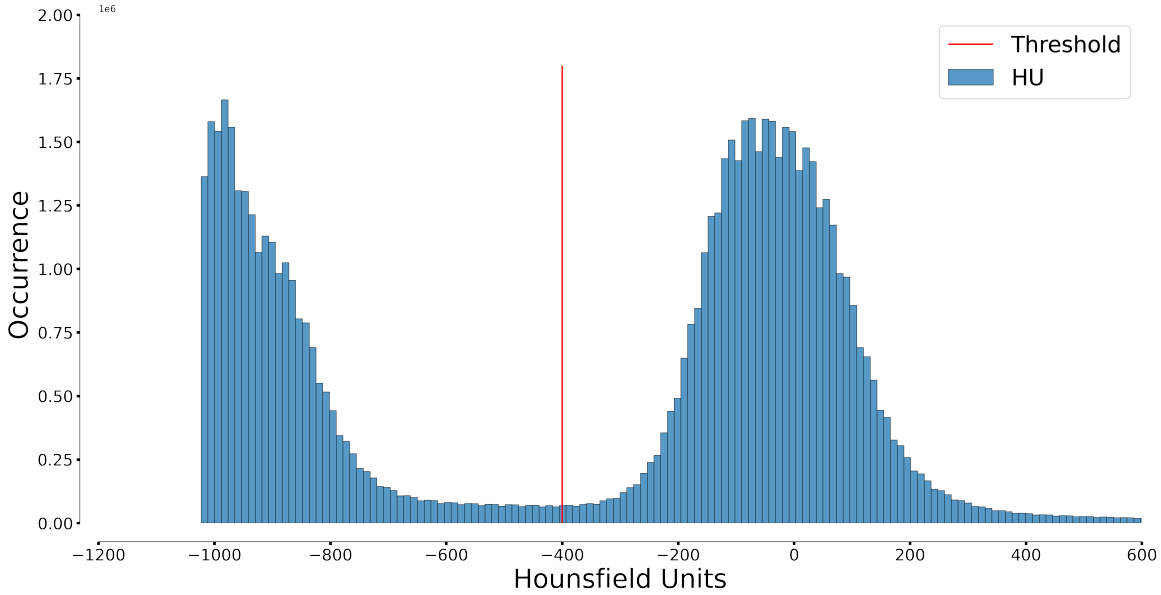



**Figure 1.1:** *Histogram of a GL image with two well delineated regions.The threshold value(red line) was set visually at -400 HU*

The threshold value is usually setting by visual assessment, but can also be automatized by algorithms like Otsu one. Sometimes may happen that more than two classes are present in the image, so we can set more than one threshold values to achieve a multi-class segmentation, also, in this case, there are algorithms to automatized this process, as an extension of the previous one called *multi Otsu threshold*.

This is a simple but very effective approach to segment images when different structures have high contrast in intensities. Threshold does not take into account the spatial characteristics of the image, so it is sensitive to noise and intensity inhomogeneity, that corrupt the image histogram and make difficult the separation [8]. To overcome these issues several variations of the threshold have been proposed based on local intensities and connectivity.

A threshold is usually used as an initial step in sequences of image processing operations, followed by other segmentation techniques that improve the segmentation quality. Since threshold uses only intensity information, it can be considered a pixel classification technique.

### 1.2.2 Region Growing Approach

Region growing approach allows extracting connected regions from an image. This algorithm starts at seed location in the image (usually manually selected) and checks the adjacent pixels against a predefined homogeneity criteria [9], based on intensity, and/or edges. If the pixels met the criteria, they are added to the region. A continuous application of the rule allows the region to grow.

Like thresholding, region growing is used in combination with other image segmentation operations, and it usually allows the delineation of small and simple structures such as tumour and lesions [8].

Regions growing can also be sensitive to noise: the extracted regions may have holes or even become disconnected. May also happen that disjoint areas become connected due to the partial volume effect.

When we use this approach we have to consider that for each region we want to segment a seed must be planted. There are some algorithms, related to the region growing, that does not require a seed point, like split and merge one. Split and merge operates recursively. The first step is to check the pixel intensity homogeneity: if they are not homogeneous, the region is split into two equal-sized sub-regions. This step leads to an over-segmentation, so a merging step is performed, which merge adjacent regions with similar intensities [9].

### 1.2.3 Deformable Model

Deformable Models use an artificial, closed, contour/surface able to expand or contract over time and conforme to a specific image feature [9]. This approach is physically motivated model-based technique for the detection of region boundaries [8].
The curve/surface is placed near the desidered boundary and it is deformed by the action of internal and external forces that act iteratively. The external forces are usually derived from the image.
This approach has the capability to directly generate closed parametric curves or surfaces from images and and incorporate smootness constraint that provides robustness to noise and spurioous edges.
However this approach requires a manual interaction to place the appropriate set of parameters.

### 1.2.4 Markov Random Field

Markov Random Field (MRF) is not a proper segmentation method, but it is a statistical model that is used within segmentation methods which model the spatial interaction between neighbouring pixels. It is often incorporated in clustering algorithms such as K-means with a Bayesian prior probability.

This model is used when most pixels belong to the same class as their neighbouring pixels, in this cases, any anatomical structure that consists of only one pixel has a very low probability of occourring [8].

A difficulty of this model is that it is very sensitive to the parameters that control the strength of the spatial interactions. The other MRF disadvantage is that it is a very computationally expansive algorithm. However, despite these disadvantages, MRF are widely used to model segmentation classes and intensity inhomogeneities [8].

### 1.2.5 Classifiers Approach

Classifiers approaches use statistical pattern recognition techniques to segment images by using a mixture model which assumes that each pixel belongs to one of a prior known set of classes [9]. To assign each pixel to the corresponding class, it

uses the so-called *feature space*, which is the space of any function of the image. An example of a 1D feature space is the image histogram.

The features of each pixel form a pattern that is classified by assigning a probability measure for the inclusion of each pixel in each class [9].

This approach assumes prior knowledge about the total numbers of features in the image and the probability of occurrence of each class. Generally, this quantity is not prior known, so we need a set of training data to use as a reference. There are different techniques which use this approach:

- **k-Nearest Neighborhood** : each pixel is classified in the same class as the training data with the closest intensity;
- **Maximum likelihood or Bayesian** : Assume that pixel intensities are independent samples from a mixture of probability distributions and the classification is obtained by assigning each pixel to the class with the highest posterior probability.

This approach requires a structure to segment with distinctive and quantifiable features. It is computationally efficient and can be applied to multichannel images. This approach does not consider spatial modelling and it requires a manual interaction to obtain the training data that must be several since the use of the same training set for a large number of scans can lead to biased results.

### 1.2.6 Clustering

Clustering approach is similar to the classifiers one, but in an unsupervised fashion, so does not require a training dataset. Clustering iteratively alternates between the image segmentation and its classes properties characterization. In this way, we can say that clustering approach trains itself by using the available data information. We can identify 3 main clustering algorith

- **k-means clustering:** partition $n$ observations in $k$ clusters iteratively computing a mean intensity for each class. It segments the image by classifying each pixel in the class with the closest mean
- **Fuzzy C-means:** this algorithm generalizes the K-means clustering in order to achieve soft-segmentation;
- **Expectation Maimization:** uses the same clustering principle as k-means by assuming that each observation belongs to a Gaussian mixture model. It is an iterative method that seek to find maximum likelihood estimates means, covariances and mixing coefficients of the mixture model.

It does not require training data, but it suffers from high sensitivity to the initial parameters and it does not incorporate spatial models: it is a pixel classification technique [8].

The most used algorithm for clustering is the k-means clustering, which seeks to assign each pixel to a particular cluster aiming to minimize the average square distance between pixels in the same cluster [12]. Each cluster is described by a vector representing its centre, so this technique is described as a centroid model [13].

The labelling is performed by assigning each point to the cluster with the nearest centroid. Each point is assigned to the cluster with the nearest centre.

Given an integer $k$ and a set of $n$ data points from $\mathbb{R}^d$, the k-means clustering seeks to find the $k$ centres that minimize a potential function given by the sum of squares:

$$\Phi = \sum_{x \in S} \min \|x - c\|^2 \tag{1.2}$$

Where $S \subset \mathbb{R}^d$ is a set of points. In this work, $\mathbb{R}^d$ will be the colours space and $S$ is the space of colour of each voxel.

The steps of the algorithm are the following:

1. Randomly initialize the values of the $k$ centres,
2. Generates the $k$ clusters associating each point to its nearest centroid
3. Re-compute the centroid for each cluster until the centroid does not change.

Arthur and Vassilvitskii [12] have pointed out that this algorithm is not accurate and can produce arbitrarily bad clusters. So they have developed a method, the "k-means++" which improves the clustering accuracy by made an accurate choice of the initial cluster centres.

They pointed out that the bad clustering is caused to the fact that $\frac{\Phi}{\Phi_{opt}}$ is unbounded even if the number of clusters and points are fixed, where $\Phi_{opt}$ is the potential function in the optimal centroids case. They proposed a variant for the choosing of the centroids: They randomly chose only the first centroid. To chose the other, they weight the initial points according to the distance square $(D(x)^2)$ from the closest centre already chosen. So the final algorithm is equal to the k-means except for the initial centroids selection that is made as follows:

1. Take one centre $c_1$, chosen uniformly at random from $S$.
2. Take a new centre $c_i$, choosing $x \in Si$ with probability $\frac{D(x)^2}{\sum_{x \in S} D(x)^2}$
3. Repeat the step 2 till $k$ centres are chosen
4. Proceed like a classical k-means clustering.

They have proved that this approach leads to better results in less time. For more details refer to [12].

### 1.2.7 Color Quantization

Colour quantization is the process of reducing the number of colours in a digital image preserving significant information. In other words, the quantization process should not cause significant information loss in the image. The colour quantizes image should be very close to the original image, visually as in Figure 1.2.

Colour quantization plays an important role in many application fields such as segmentation, compression, colour texture analysis, watermarking, text localization/detection, non photorealistic rendering and content-based retrieval [14].

Colour quantization is used for image segmentation. To this purpose the algorithm aims to reduce the number of colours in the image to one of the different

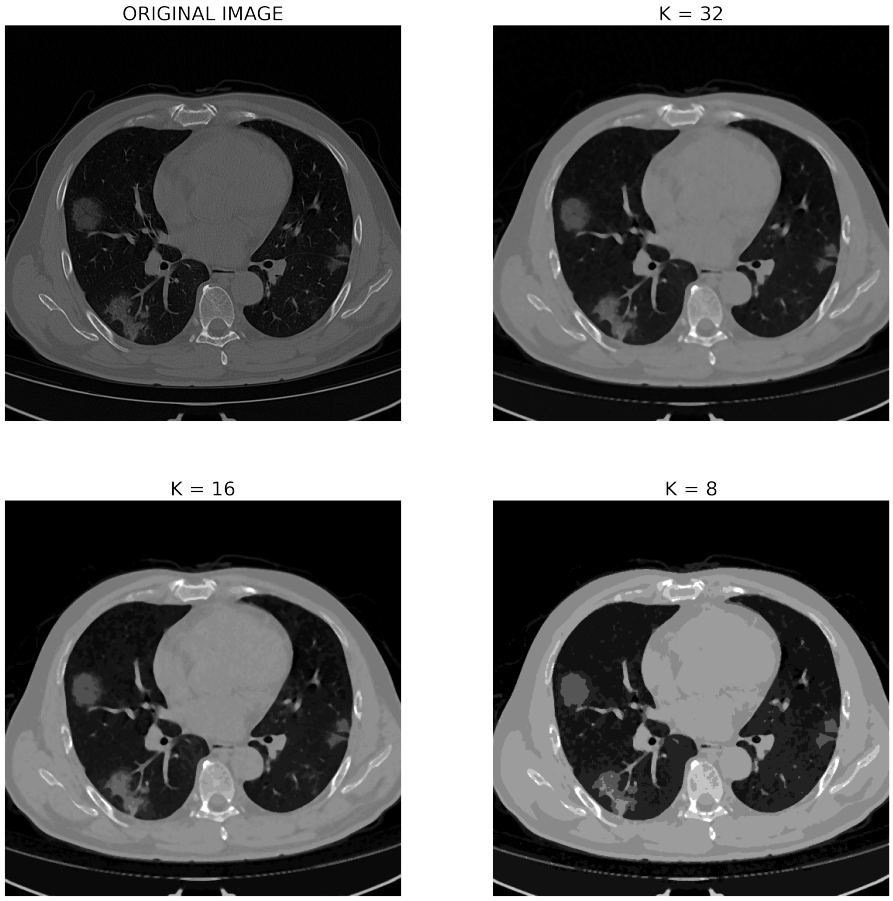



**Figure 1.2:** *Colour quantized GL image. We observe the original image, a 32 colour image which looks similar to the original one, a 16 colours image and 8 colours image. The tissues are grouped by colour similarity. Reduce the colour to the number of the cluster allows pixel classification.*

tissues, assigning a characteristic colour to each one of them. This is the case of medical image segmentation, in which each image colour represents a particular characteristic of the tissue displayed (i.e in x-ray represent$\mu$). To perform this technique, different algorithms may be used to group the colours, like clustering algorithm or the principal component analysis.

### 1.2.8 U-Net

Artificial Neural Networks(ANN) are a computational architecture derived from neural physiological models [9]. This architecture is made by interconnected artificial neurons able to perform elementary operations. For imagery, analysis is usually used Convolutional Neural Networks(CNN), also known as shift invariant or space invariant artificial neural networks (SIANN).

In biological image processing, the so-called U-Net is usually used. U-Net is a kind of convolutional neural network which allows overcoming the requirement of many training data. That because large training dataset is not always available like often happens in medical and biological segmentation purposes.

Convolutional Networks are usually applied to classification tasks. In biological and medical purposes, the segmentation should include localization and a class label

should be assigned on each voxel. To achieve these purposes, in 2015 for the ISB cell tracking challenge, Olaf Ronneberger, Philipp Fischer, and Thomas Brox have developed this kind of network [15].

The U-Net is able to work with only small set of training samples. In order to work with few training data,we have to make a huge data augmentation, by applying an elastic deformation to the training images. The whole structure is divided into two main parts:

- Contraction path (*encoder*) : sequence of convolutional and pooling layers, which aim to extract features and reduce the input dimensionality.
- Expansion Path (*decoder*) : second set of convolutional and up-sampling layers, to reconstruct the feature map and consequently the segmentation mask, which aims to process the extracted features

The contractive path and the extractive path. The contracting path is a typical convolutional network that consists of repeated application of convolutions, each followed by a rectified linear unit (ReLU) and a max-pooling operation. During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path [15].

Decode path tends to lose some of the higher-level features that the encoder learned: Using shortcut connection, the output of the encoding layers are directly passed to the decoder layer, preserving the important features [16].

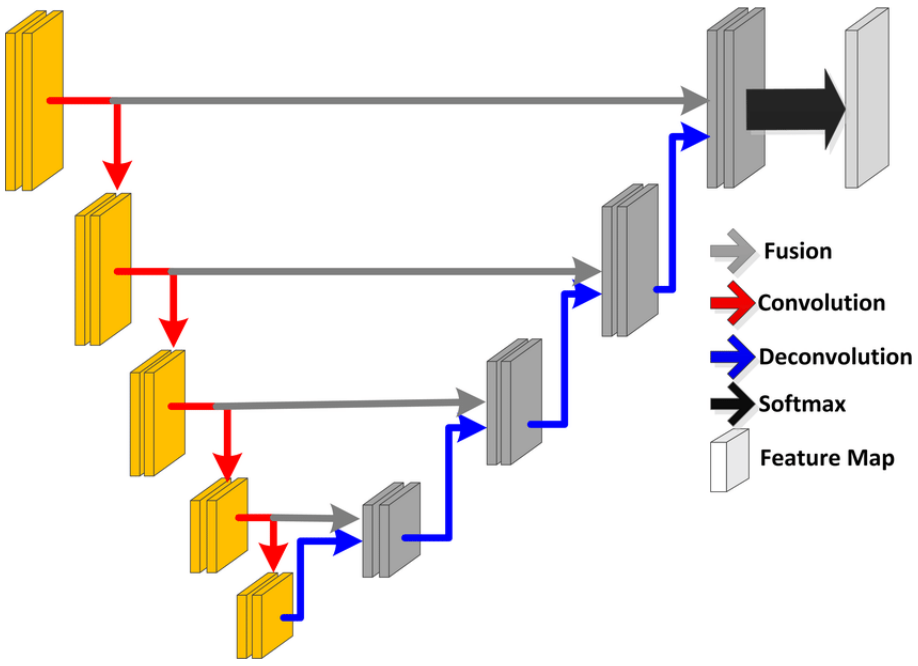



**Figure 1.3:** *U-Net network architecture: We can see the U-shape made by symmetricity between expansion and contraction path. The grey arrows indicate the shortcut used to prevent information loss.*

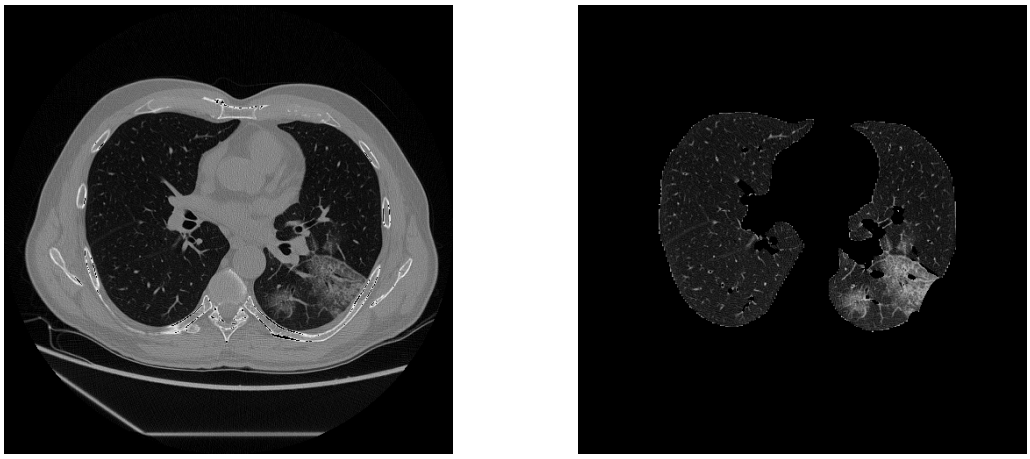
# Chapter 2

# Pipeline

The aim of this work of thesis is the developing of an automatic pipeline for the segmentation of Ground Glass Opacities and Consolidation areas in chest CT scans of patients affected by COVID-19. Austin, in *Glossary of terms for CT of the lungs* [17] has defined the GGO and CS as:

"*Hazy increased attenuation of lung, but with preservation of bronchial and vascular margins; caused by partial filling of air spaces, interstitial thickening, partial collapse of alveoli,normal expiration, or increased capillary blood volume, which is different from consolidation in which bronchovascular margins are obscured.*

.



**(a)** *Chest CT scan with GGO and CS* **(b)** *Segmentaed lung regions*

**Figure 2.1:** *Original Chest CT scan(a), we can clearly see the GGO and CS regions. Moreover, all extra -lung regions are present. In (b) we can see the same slice but after the lung segmentation. We can see that each different region has a similar grey level.*

The starting point is the chest CT scans (Figure 2.1a). Firstly I have removed all the extra lung regions like body, bronchial structures and CT tube which are a potential source of false positives. Once we have obtained a scan as Figure 2.1b , we can notice that the different structures are characterized by similar grey level:

the basic idea was to use the colour quantization for medical image segmentation, grouping voxels based on colour similarity, and assigning to each tissue a characteristic colour. This can be done since in CT scan exists a relationship between the tissue in the voxels and the Gray Levels used to display it, given by the Hounsfield Units(eq 1.1): the colours are proportional to HU, which are defined as a linear transformation of the linear attenuation coefficient($\mu$).

We can consider different properties beside the single voxel intensity. As we can see, lesion areas involve many closest voxels: It is interesting to incorporate also neighbourhood voxel information in the colour quantization. Moreover, the contrast between sick and healthy areas may change according to the severity of the disease, so we can incorporate different gamma of the image, to enhance these regions.

Adopting a colour model for the digital image allows encoding colour information in the pixels. Colour can be used to encode also voxel neighbourhood information assigning to each channel a different function of the image.

Once I have, build the colour space. I have to find the characteristic colour of each tissue under study, which is represented by a centroid in this space. I have created a training dataset made of CT scans from different patients and applied a k-means clustering since it provides a good balance between segmentation performance and computational efficiency. K-means clustering requires prior knowledge about the number of clusters, which in our case is given by the anatomical structure of the lung. I have considered a cluster for each different anatomical structure:

- Lung;
- Edges;
- Vessel surrounding bronchial structures;
- Bronchi.
- Ground Glass Opacities and Consolidation;

Once I have estimated the centroids for each tissue, I have used them for the segmentation: each voxel of the image is assigned to the cluster of the closest centroids. The final pipeline structure is summarized in Figure 2.2.

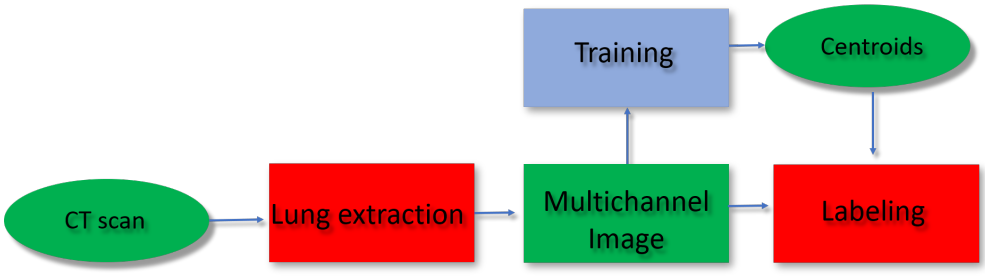



**Figure 2.2:** *Workflows of the developed pipeline. CT scans are taken as input and a lung segmentation is performed. After that, there is the construction of the multichannel image. The labelling step takes as input the multichannel image and the centroids, previously estimated by a training.*

Except for the lung segmentation, the learning process for the GGO and CS segmentation is unsupervised. Once the centroids are estimated, the training step

does not need to be repeated. Moreover, in the implementation, the building of the multichannel image is incorporated in the labelling step. As a consequence the final pipeline structure looks like in Figure 2.3.

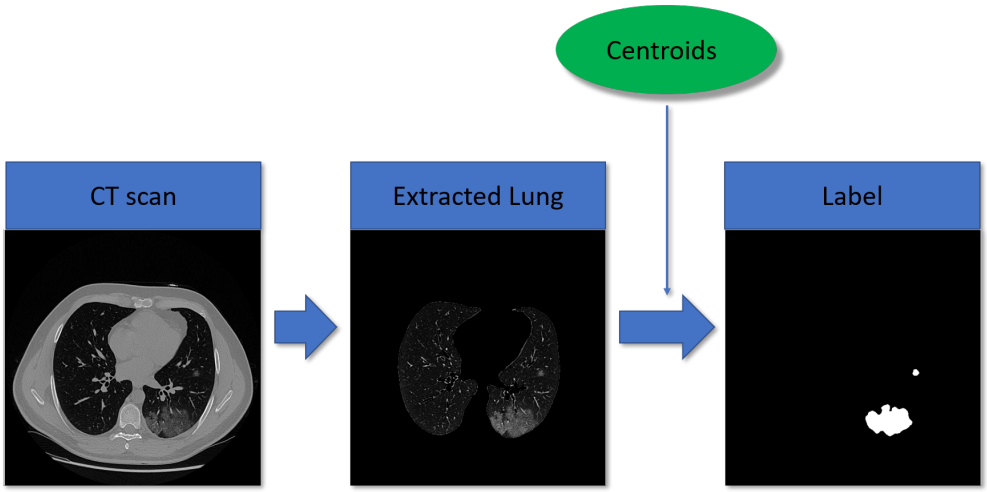



**Figure 2.3:** *Final pipeline structure, from left to right we can see the input image stack, the isolated lung regions and the final label. To performed the labelling a set of pre-computed centroids was used.*

## 2.1 Description

As we have seen, the pipeline workflow is divided into many blocks, each one performing a different task. In this section, I will describe in details how each step of the pipeline is achieved. This section involves only a description. The implementation will be discussed in the next one.

### Lung Extraction

This preliminary step is performed before both training and labelling and involves the managing of the HU, the isolation of lung regions and the removal of the main bronchial structures. The registration of the HU on a common space is necessary to overcome the issues that may arise from the different padding values used by the different manufacturer of the CT scans. Moreover, some noise may change air values of some regions, which will be not exactly $-1000$. During the scan acquisition, all the regions outside the CT tube aren't sampled, so to obtain a square $N \times N$ image for each slice some padding values are added, which different values according to the scan manufacturer. This registration allows overcoming some issues that may arise during the conversion of the image from 16 to 8 bit unsigned integers, required to operate with OpenCV functions.

Lung segmentation is a pivotal pre-processing step in many image analyses such as identification and classification of lung pathologies [15]. The lung isolation allows to find a mask for the lung regions, and thus excluding all the body regions, the CT tube and the extra-lung organs like intestine and heart, avoiding the formation of false positives.

Automatic lung segmentation algorithms are typically developed and tested on limited datasets and usually over a limited spectrum of visual variability by containing mainly cases without severe pathologies [15]. Rule-based approach, like thresholding, region growing, etc., usually fails for CT scans of patients with severe Interstitial Lung Disease (ILD), as we can see in Figure 2.4. For these reasons to achieve the lung segmentation, I've used a pre-trained U-Net [15] [18].

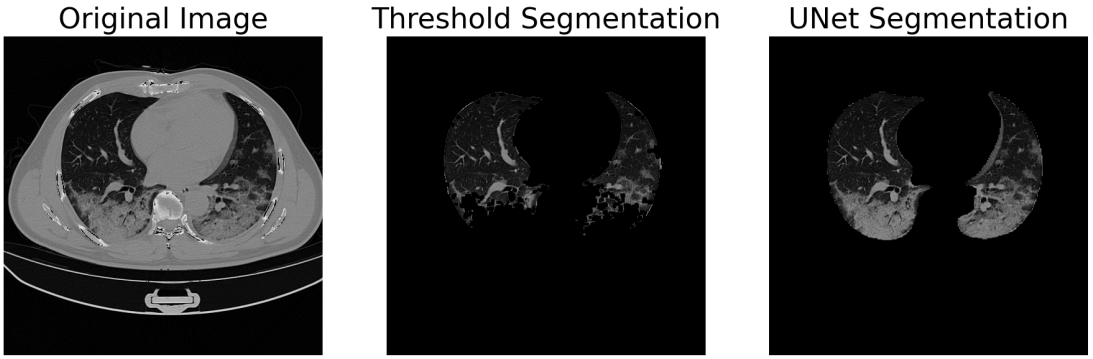



**Figure 2.4:** *From left to right the original CT scan of a patient with severe ILD, the lung segmented by threshold and it's connected components, and lung segmentation achieved by the pre-trained U-Net. We can clearly see the missing areas in the first segmentation, correctly identified in the U-Net one.*

This kind of segmentation includes in the lung region also motion artefacts and bronchial structures, which are the main source of false positives. To achieve a better segmentation a refinement process is performed, which aims to remove the main bronchial structures form the selected lung regions.

Bronchi have elongated shape respect to the other structure which usually is rounded: the basic idea was to use this kind of information. To perform this task, I have computed the covariant matrix of the derivative in a neighbourhood (2.1) and the corresponding eigenvalues.

$$M = \begin{bmatrix} \sum_{S(p)} (dI/dx)^2 & \sum_{S(p)} dI/dx dI/dy \\ \sum_{S(p)} dI/dx dI/dy & \sum_{S(p)} (dI/dy)^2 \end{bmatrix} \tag{2.1}$$

If a particular region has an elongated shape, one of the eigenvalues (corresponding to the eigenvector in the direction of the structure) will be higher than the other, otherwise, both eigenvalues have a low one. I have applied this filter on each slice of the scans and took a map of the maximum eigenvalues.

In Figure 2.5, I've displayed the image after the lung segmentation by the neural network and the corresponding eigenvalues map. As we can see the higher values of the map correspond to the edges of the main bronchial structures. To create the mask for these structures, I have applied a fixed threshold to the map. Since the main bronchial structures are large, this process can remove only part of the edges, but the inner structure is preserved. To refine the segmentation, this process is repeated a second time, allowing a fine removal of the structures.

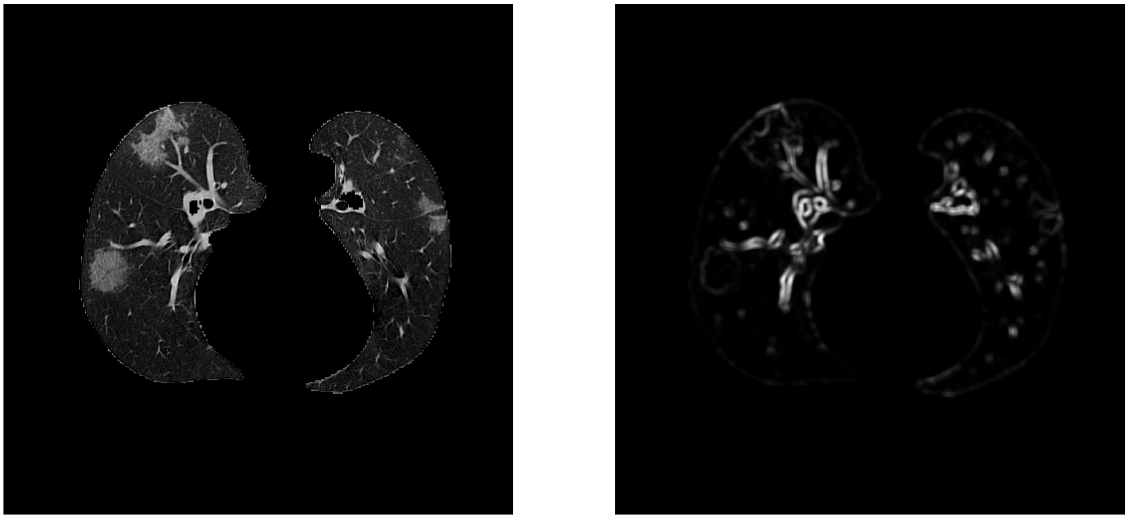


**Figure 2.5:** *From left to right: lung regions selected by the U-Net, maximum eigenvalues map of the lung. As we can see the U-Net does not exclude the bronchial structure from the lung, on the other hand, the maximum eigenvalues map delineates very well these regions. I have used this map to remove the unwanted bronchial regions.*

## Multi Channel Image

This step involves the preparation of the images, with the computation of the different functions and the building of the multi-channel image that incorporates neighbouring and edges information. The used image is composed of 4 channel built as follows:

- Pure image after Contrast Limited Adaptive Histogram Equalization (CLAHE), with a block size of $10 \times 10$ pixels
- Image after a median blurring with kernel size equal to 11 pixes
- Image after a gamma correction with $\gamma = 1.5$
- Standard filtered image with a kernel of size 3 pixels

In Figure 2.6 I have displayed the four different channels of the image. Each channel allows us to consider different information.

The histogram equalized and the gamma-corrected images allow to take into account information about the single voxel. The histogram equalization is applied to enhance the image contrast by improving GL usage. For each slice, the histogram is equalized considering only a $10 \times 10$ area, to take care of the over-amplification of the contrast.

The gamma correction is a non-linear operation and is used to decode the luminance and to make in evidence the low contrast regions.

The median blurring allows us to consider also the information about the neighbourhood voxels, allowing the reduction of the outliers. The usage of the filter is justified since the lesions involve several closest voxels.

The last channel consists of the image after the application of a local standard deviation filter. This filter consists of the replacement of each pixel value with the

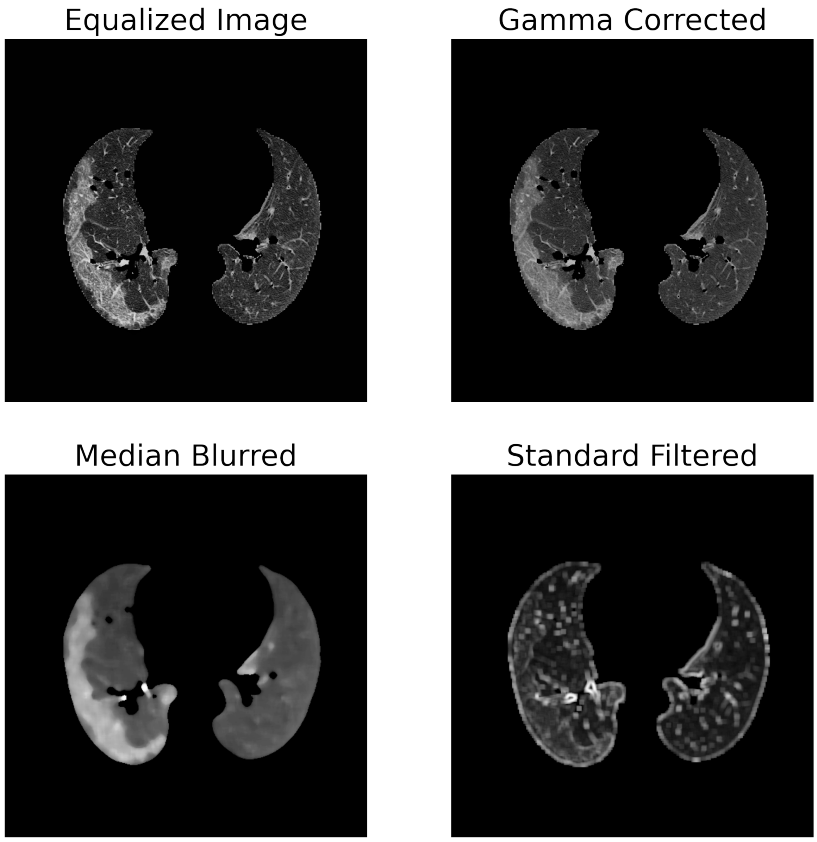



**Figure 2.6:** *Channels of the image. From left to right and from top to bottom the image after the histogram equalization, the gamma correction, the median blurring and the std filtering. These channels allow us to consider information about single voxel, neighbouring voxels and their variability. The histogram equalization is applied over small* $10 \times 10$ *areas to avoid over-amplification of the contrast. The gamma correction was performed using* $\gamma = 1.5$*, and the median and std filter kernel sizes are respectively* 11 *and* 3*. Each image is normalized according to the mean and standard deviation of the whole scan.*

standard deviation of its neighbourhood. It helps us to distinguish the bronchial structures and motion artefacts not removed during the lung segmentation.

Each channel of the image is normalized according to means and standard deviation of the whole scan; because of the k-means clustering.

## Training

This step involves the estimation of the centroid set. To achieve this purpose, k-means clustering was performed on a training set made on multichannel images from many CT scans. During this task, I have to take into account that k-means clustering requires a balance in the different cluster representation. As we can see in Figure 2.7, the main part of the image is composed of background, that is an over-represented cluster. Moreover, we can observe that the amount of involved lung volume may change according to the severity of the disease: the first scan presents a low amount of GGO and CS; the second one has a larger lung region involved.

The images of the training set are then shuffled and split into many subsamples

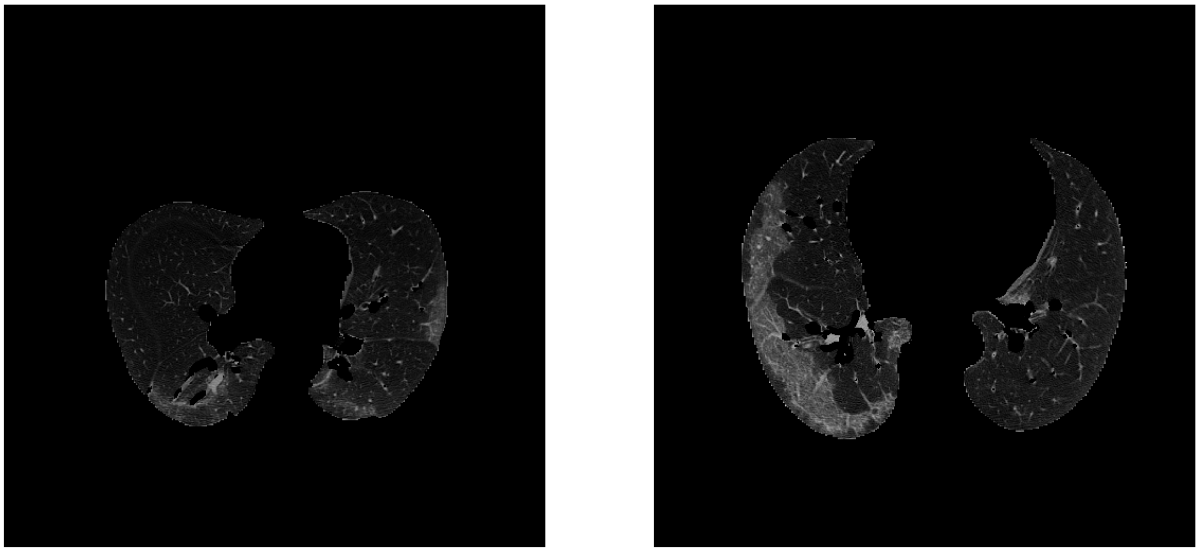


**Figure 2.7:** *Lung regions with different ground-glass areas. We can notice that the main part of the image is composed only by the background. From left to right respectively a scan with small lesion areas and one with a large lesion.*

and the clustering is performed independently on each one of them. As a final centroid set, I have considered the one that minimizes the intra-cluster variance. This step is justified since the results of the k-means clustering depend on the initial choice centroids. During the minimization of the potential function, the algorithm may find a local minimum rather than the global one. The different clustering allows to estimate more than one possible solution and to choose the best one. Moreover, the creation of several sub-samples is made since the creation of a single, large array with several images is not always possible, since requires a huge quantity of memory to be allocated.

## Labelling

This step requires as input the multichannel image and a pre-estimated centroid set. It involves the assigning of each voxel to the cluster corresponding to the nearest centroid and the selections of the one corresponding to GGO and CS areas. In this way we are performing a pixel classification by assigning the regions to a particular label according to their intensities information: this allows us to group on the same cluster objects that are spatially disconnected as often happen in the medical imaging field.

The distance between voxel colours and each centroid is defined as the euclidean distance:

$$d(x_j, c_i) = \sqrt{(x_j - c_i)^2}$$

Where $x_j$ is the color vector for the $j_{th}$ voxel and $c_i$ is the $i_{th}$ centroid.

## 2.2 Implementation

I have implemented the pipeline described before using python, which is an high level object oriented programming language. To perform all necessary operations (image filtering, input/output managing, etc.) I've used three libraries: OpenCV [19] , SimpleITK [20] and Numpy [21].

The whole code is open source and available on GitHub [22] and the pipeline installation is automatically tested on both Windows and Linux by using AppveyorCI and TravisCI. The installation is managed by setup.py, which also provides the full list of dependencies. The code documentation was generated by using sphinx and it is available online (`https://covid-19-ggo-segmentation.readthedocs.io/en/latest/?badge=latest`).

The pipeline provides 3 scripts: lung_extraction, labeling and train, to perform the image segmentation and the centroid estimation. However, also, a pre-trained centroid set is provided. Till now I've used the (silent) hypothesis that the segmentation involves only one scan. To automatize the segmentation on several CT scans, PowerShell and bash script are provided.

An important point is the input and output managing. The pipeline operations involve only the voxel array, but the input image formats provided also additional information like voxel spacing, image origin and direction. This information must be preserved and incorporated in the output since are necessary to preserve the compatibility with medical image processing software and medical image format. To achieve this purpose, I have implemented input and output method based on the SimpleITK IO interface: as a consequence, the supported image formats are the one supported by SimpleITK. The input method allows to read an image in the medical image format and get the voxel array and the spatial information. The output one allows saving the image in a medical image format by setting also voxel array and spatial information.

The construction of the multi-channel image is embedded both in training and labelling script. This step requires the application of 4 filters on the image and to stack the together the results. For the image filtering, I've used the corresponding function implemented in OpenCV. These functions can be applied only on a single image, so I have applied them slice by slice along the axial direction. In this way each image is processed independently from the others: No 3D structure is exploited. To stack the images, I've simply used the Numpy stack method. Notice that the OpenCV functions works only with 8−bit grayscale images. The input image must be converted from HU to GL image. This step is performed at the end of the lung extraction.

### 2.2.1 Lung Extraction

This script aims to achieve the HU registration, the lung segmentation and the bronchial removal. It takes as input the CT scan in each format supported by SimpleITK. As output will return the stack after the lung extraction, saved as *nifti*. Its workflow is summarized in1.

The lung mask was created by applying the pre-trained U-Net by using the suitable method from [18] with the pre-trained model R231 [15].

After the creation of the lung mask, we have to achieve the registration of the

**Algorithm 1:** Lung Extraction

**Data:** Volume (CT scan)
**Result:** Volume with extracted lung

mask←ApplyUNetVolume(Volume)
volume←(volume < -1000) =-1000
volume←ShiftMinimumTo1(volume)
/* Start the bronchial removal */

eigen← maxEigenvalues(lung)
bronchial_mask← (eigen < threshold)
lung_woBronchi← (lung ∘ bronchial_mask)
/* Refine the bronchial mask */

eigen←maxEigenvalues(lung_woBronchi)
bronchial_mask←(eigen < threshold)
lung_woBronchi← (lung_woBronchi ∘ bronchial_mask)

HU on the whole scan. This takes care of the padding values and of the noise, that may raise at values less than $-1000\,HU$. All the values less than air one are set to $-1000$, after that all the unit are shifted to zero. Notice that set the padding values equals the air value does not raise problems, since they are removed after the application of the mask. Moreover shifting all the values to 0 simplify the application of the mask. These operations are performed on the image tensor by using NUMPY ndarray method. After each value is rescaled in $[0, 255]$.

Listing 2.1: HU registering function

```
1
2  import numpy as np
3
4  def shif_and_crop(tensor) :
5
6    tensor[tensor <  -1000] = - 1000
7    tensor = tensor + 1000
8
9    return tensor
10
```

More interesting is the estimation of the eigenvalues map. To achieve this purpose I've implemented a function based on CV2.CORNEREIGENVALSANDVEC in OPENCV. This function takes as input an 8−bit grey scale image, so we have to rescale the GL value of the image tensor.

In the end the computing of the eigenvalues map is made by the following function:

Listing 2.2: maximum eigenvaleus map

```
1
2  import cv2
3  import numpy as np
4  from functools import patial
5
```

```
6   def max_eigenvalues_map(tensor, block_size, kernel_size) :
7
8     func = partial(cv2.cornerEigenValsAndVecs,
9               blockSize = block_size,
10              ksize = kernel_size)
11    res = np.array(list(zip(*list(map(func, tensor)))))
12    res = res.transpose(1, 0, 2, 3)
13    res = res[:, :, :, :2]
14    max_eigenvals = np.max(res, axis = 3)
15
16    return max_eigenvals
17
18
```

Once the required parameters were set, the OpenCV function is iterate over the axial slices. The following transposition and selection of tensor elements are needed since the function returns also the eigenvectors, in which we are not interested in.

At the end of the whole procedure, the resulting tensor is saved into a medical image format.

### 2.2.2 Training

This script allows for estimating the set of centroids. It requires as input a path to the training dataset. Each training scan must be saved in a format supported by SimpleITK. This script will return as output the set of centroids in *.npy* format.

This script will read the training dataset, construct the multichannel image, create the subsamples and cluster them independently. Notice that during the clustering process the background must be removed. This script is summarized in 2.

**Algorithm 2:** training

**Data:** CT scans with Extracted lung
**Result:** Centroid matrix

**foreach** *scan* ∈ *input_scans* **do**
  read the scan
  sample←image_array
**end**

/* prepare subsamples */
sample← build_multichannel(sample)
sample←shuffle(sample)
subsamples←split(sample)
/* start the first clustering */
**foreach** *Sub* ∈ *subsamples* **do**
  center←kmeans(sub, number of centroids)
  centroid_vector, internal_variance←append(center)
**end**

/* Refinement */
centroid_matrix←centroid_vactor(min(*internal_variance*))

The clustering process is managed by the KMEANS_ON_SUBSAMPLES function, implemented as follows.

Listing 2.3: kmenas_on_subsamples

```
1
2  import cv2
3  import numpy as np
4  from tqdm import tqdm
5
6  def kmeans_on_subsamples(imgs,  n_centroids, stopping_criteria,
7               centr_init, weight = None) :
8
9    if weight is not None :
10     vector = np.asarray([ el[w != 0]
11               for el, w in zip(imgs, weight)],
12               dtype = np.ndarray)
13   else :
14     vector = np.asarray([el.reshape((-1, el[-1]))
15                 for el in imgs],
16                 dtype=np.ndarray)
17
18   centroids = []
19   ret = []
20   for el in tqdm(vector) :
21
22     r, _, centr = cv2.kmeans(el.astype(np.float32), n_centroids,
23                 None, stopping_criteria,
24                 10, centr_init)
25     centroids.append(centr)
26     ret.append(r)
27
28   final_center = centroids[np.argmin(ret)]
29
30   return np.asarray(final_center, dtype= np.float32)
31
32
```

This function takes as input the array containing the subsamples and apply k-means clustering on each of them, storing all the values. To remove the background, it uses a weight tensor. The values of this tensor must be 0 for the voxel corresponding to the background, and 1 otherwise. At the end of the procedure, the centroid set which provides the minim value of the sum of square error is selected.

### 2.2.3 Labelling

This is the last step of the pipeline and involves assigning of each voxel to the cluster corresponding to the nearest centroids, in this way a hard segmentation is achieved.

The script takes as input the CT scan after the lung extraction and it builds the multichannel image as described before. After that it assigns each voxel to the cluster of nearest centroids, which is the one that minimizes the distance:

$$cluster = \arg\min_{S} \sum_{i=1}^{k} \sum_{S} \|x - \mu_i\| \tag{2.2}$$

where $x$ is the colour vector of the voxel and $\mu$ is the $ith$ centroid. During this process, the background is automatically assigned to the 0 labels, passing a mask

which assumes 0 on the voxels background and 1 for the other one. At the end of the assignment, only the cluster corresponding to GGO and CS is selected.To summarize the process, the pseudocode of the script is reported in the algorithm,3. I have tested this algorithm on three different datasets. The results are described in the next chapter.

**Algorithm 3:** Pseudo-code for the labeling script

**Data:** CT scan to label, centroids
**Result:** GGO label
image←build_multi_channel
/* Compute distances and found the minimum */
**foreach** $c \in centroids$ **do**
  distances← $\|image - c\|^2$
**end**
labels← arg min $(distances)$

The assignment process is performed by the IMLABELING function, which takes care to assign the background to 0, if the suitable parameter is passed. The function is implemented as follows:

Listing 2.4: imlabeling

```
1
2  import numpy as np
3
4  def imlabeling(image, centroids, weight = None) :
5
6
7    if weight  is not None :
8      distances = np.asarray([np.linalg.norm(image[weight != 0] -c,
9                  axis = 1) for c in centroids])
10
11     weight[weight != 0] = np.argmin(distances, axis = 0)
12     return weight
13   else :
14     distances = np.asarray([np.linalg.norm(image -c, axis = 3)
15                 for c in centroids])
16     labels = np.argmin(distances, axis = 0)
17     return labels
18
19
20
```

## 2.3 Optimization

During each step of the pipeline, we have to set different parameters, like the kernel size for median and std filter, as well as the number of centroids to use for the segmentation. In this section, I will briefly describe how each one of these parameters was optimized, to obtain the best segmentation performances.

### 2.3.1 Number of Clusters

The designed algorithm for the centroids estimation is the k-means clustering that requires prior knowledge about the number of clusters to use. This is very important since a bad choice will badly affect the whole segmentation results. In order to choose the proper number of clusters, I've considered two different sources of information: the anatomical knowledge about the lung and the internal variability of the lung.

From anatomical knowledge about the lung, we can derive 5 clusters, corresponding to:

- Lung;
- Edges;
- Vessel surrounding bronchial structures;
- Bronchi.
- Ground Glass Opacities and Consolidation;

Notice that the background of the image is not considered as a cluster since it is removed from the segmentation for the reasons explained before. In order to verify that this number of clusters is the best one, I have considered the internal cluster variability.

Clustering techniques try to group the data into different clusters in order to maximize the difference between points in different clusters and to maximize the similarity within each cluster. If the number of centroids is less then the clusters one, the similarity within each cluster is low. Increasing the number of centroids will reduce the internal variability until 0 (if the number of clusters is equal to the number of points). This means that after a certain point the diminishing of the internal variability is no more significant, since do not correspond to the good number of clusters but only to the increasing of their number.

To found the correct number of clusters, we seek to for a number of clusters which still provides a small amount of internal variability.

To achieve this purpose, the clustering was repeated several times increasing the number of clusters and for each iteration, the internal variability was measured by the sum of squares estimate error (SSE) :

$$SSE = \sum (x_i - c_j)^2 \tag{2.3}$$

Once this task is completed, the results was printed in Figure 2.8.

The optimal number of clusters is the one that corresponds to the elbow of the curve. It is difficult to find this feature from visually, I took the numerical value of the elbow considering the point which maximizes the distance between the right joining the first and the last point. From this analysis, I have found that the optimal number of clusters is 5, which corresponds to the same that I have found considering lung anatomy.

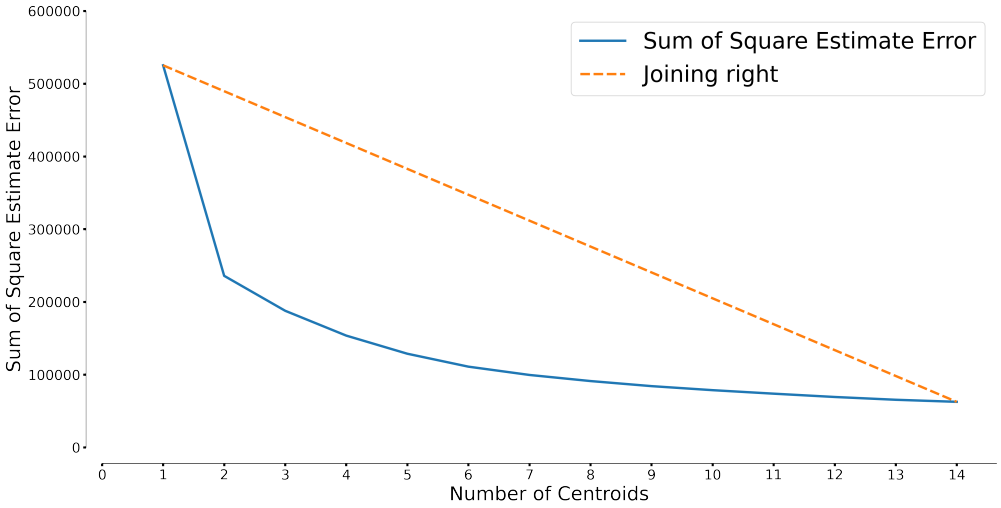



**Figure 2.8:** *Intra-cluster variance vs the number fo clusters(blue); right joining first and last point(orange). We can see the elbow shape of the graph; the elbow location in the one which maximizes the distance between the right joining line and the elbow curve.*

### 2.3.2 Kernel Size

During the building of the multi-channel image, I had to compute different image features, that requires the setting of different parameters, like median or standard filter kernel sizes. To achieve the best segmentation, I have performed an optimization step that aims to find the parameters that allow obtaining the best segmentation.

Notice that this process is not necessary and a good segmentation can be achieved also by setting these parameters manually.

In order to perform the optimization, I have used SCIKIT-OPTIMIZE [23], more specifically the GC_MINIMIZE METHOD.

This method seeks to perform a Bayesian optimization by using a Gaussian process to approximate an objective function. The function values are assumed to follow a multivariate Gaussian. The covariance of the function values is given by a GP kernel between the parameters. Then a smart choice to choose the next parameter to evaluate can be made by the acquisition function over the Gaussian prior which is much quicker to evaluate [23].

In the end, an objective function is minimized. I have used the objective function defined in 4 in which is also reported the whole minimization pseudocode. The used function aim to minimize $1 - IoU$ where the IoU(Intersection over Union) is computed between the output labels and a reference one:

$$IoU = \frac{Area\,of\,Overlap}{Area\,of\,Union}$$

I have decided to use the IoU instead of accuracy since the number of pixels concerning the labelled object is very few against the number of pixels related to the background. Thus, the label would be a matrix with a large number of zeros (background) and only a few ones (object). In this case, the standard metric functions have to consider an unbalanced number of samples so the solution was to use the IoU which measures the ratio between the Intersection and Union of the output labels and the binary ground truth [16]:

**Algorithm 4:** Parameter Optimization Algorithm

**Data:** Test scans, Ground Truth
**Function** `objective`($parameters, ref\_labels, CT\_scan$)`:`
  $labels \leftarrow$ segment($CT\_scan$)
  iou = IoU($labels$, $ref\_labels$)
  **return** $1 - iou$
$best_parameters \leftarrow$gc_minimize(objective, n_calls, n_random_init)

This process allows optimizing the parameters to obtain better results. As reference labels, I have used the ones evaluated as gold standard from five experts radiologists at least 2 years of experience.

This procedure allows only to tune the parameters for better segmentation, but the learning process remains unsupervised.

# Chapter 3

# Results

The developed pipeline was trained and tested on three datasets (Sant'Orsola, ZENODO and MOSMED), described in the first section. The main method for the evaluation was a quantitative comparison with a gold standard segmentation, made and validate by 5 experts with at least 2 years of experience. In collaboration with Sant'Orsola, the pipeline results were compared with some semiautomatic segmentation by a blind evaluation made by experts.

## 3.1 DataSet Description

This section is dedicated to the description of the datasets used for the developing and test for the pipeline. The description includes general image characteristics and some metadata. If within the datasets are provided also some manual annotation, also the segmentation masks are described.

### 3.1.1 Sant'Orsola

Sant'Orsola data were the ones mainly considered in this work. It consists of 83 anonymized CT scans from 83 different patients affected by COVID-19 and 8 scans from healthy controls. Within these scans, also manual annotations were provided. These annotations were obtained with a semi-automatic approach. The built of each annotation may require several hours.

The series are distributed as follows:

| **Property** | **Value** |
|---|---|
| Number of Scans | 83 |
| Distribution by sex(M/F/O) | 66.3/33.7/0 |
| Distribution by age(min/median/max) | 35/60/89 |

For 5 scans were provided also other semi-automatic segmentations. These segmentations were obtained by refining the initial segmentation of certified software. The building of these labels has required several days. In the end, these results are validated by 5 experts with at least 2 years of experience. This 5 segmentation represents the gold standard used as ground truth.

### 3.1.2 MOSMED

MosMed [7] is a dataset which contains 1110 anonymized CT scan of the human lung from both patients affected by COVID-19 in several stages of the disease and healthy controls. A small subset of these scans is labelled. The scans are obtained between 1st March and 25th of April 2020 by different Russian hospitals. This dataset was born with educational and AI developing purposes. The studies are divided into five categories, from healthy patients to the most severe cases. Each scan of the dataset is saved in *.nfti* format and during the conversion from the original dicom series only 1 image every 10 was preserved [7]. The resulting dataset has the following characteristics:

| **Property** | **value** |
|---|---|
| Number of Scans | 1110 |
| Distribution by sex(M/F/O) | 42/56/2 |
| Distribution by age(min/median/max) | 18/47/87 |
| Number of studies in each cathegory | 254/648/125/45/2 |

The CT scans are organized into five categories, depending on the percentage of the involved lung volume:

| **Class** | **Description** |
|---|---|
| CT-0 | Normal lung tissues |
| CT-1 | presence of GGO, lung parenchima involved less than 25% |
| CT-2 | GGO, involvement of lung parenchima in $25 - 50\%$ |
| CT-3 | GGO and consolidation, involvement of lung parenchima in $50 - 75\%$ |
| CT-4 | GGO, consolidation and reticular changes, lung parenchima involved more than 75% |

Of these five categories, only 50 annotations are available, mostly involving only the patients of CT-1 groups. Scans have been annotated by the experts of Research and Practical Clinical Center for Diagnostics and Telemedicine Technologies of the Moscow Health Care Department [7].

### 3.1.3 ZENODO

This dataset consists of 20 CT scans of patients affected by COVID-19, labelled by two expert radiologists and verified by the third one. The anatomic structures labelled are the left and right lung and the infections regions [6]. Each file is in nifti format and no patient metadata was available (Exept fro spatial information like voxel spacing, origin and direction).

Unfortunately, only half of the scans are in HU, the remaining are in 8-bit grayscale, which is not suitable to verify the pipeline since requires it as input an image in HU.

## 3.2 Timing

One of the highlights of the pipeline is the brief amount of time required to obtain a complete segmentation. To characterize these features, I have performed a

benchmark on 5 CT scans with a different number of slices. For each scan I have performed separately the lung extraction and the labelling step, repeating the procedure several times. The whole test was performed on the server of the Department of Physics and Astronomy.

As we have seen, the pipeline operates on the single slice and it is repeated for the whole scan in the axial direction. As a consequence, the time required to segment different scans changes linearly according to the number of slices. To measure the time required to segment a single slice, I have performed a linear fit on the results of the benchmark and I have computed the angular coefficient.

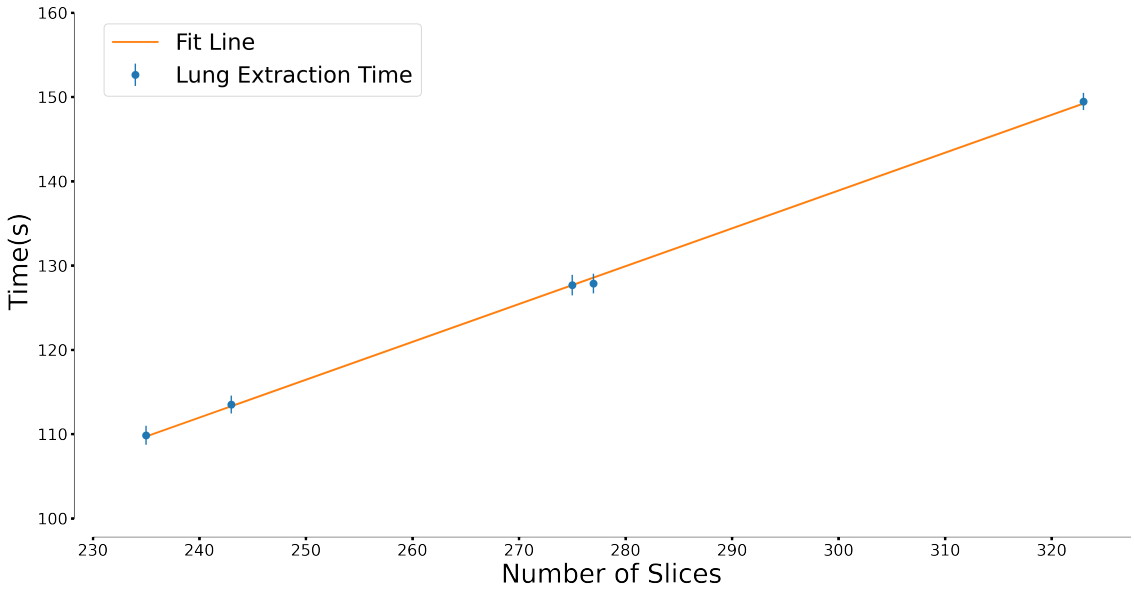



**Figure 3.1:** *Lung segmentation time vs the number of slices. We can see that the time required to perform the lung segmentation increase linearly with the number of slices. We can also see that in the worst case the lung extraction requires more or less* 2 *minutes.*

In Figure 3.1 I have reported the time for the lung extraction as a function of the number of slices in the scan: as we expected it has a linear trend. In the worst-case, It requires $150\,sec$ to perform the lung extraction. With the linear fitting, I was able to find the time required to process each slice which results: $449.94 \pm 0.03\,ms$. We have to notice that these times are acquired without GPU support. Since the application of the U-Net model is performed by TORCH , with suitable hardware it is possible to parallelize the computing and achieve also lower times.

In Figure 3.2 I have reported the results for the labelling. As we can see also, in this case, the trend is linear and in the worst case requires less than $17\,sec$. As before I have measured the time required to label each slice which results: $45.65 \pm 0.05\,ms$.

In the end, we have seen that the pipeline achieves segmentation in a low amount of time, which can be further reduced if proper hardware is used.

## 3.3 Accuracy

In this section, I will discuss the results achieved by the implemented pipeline, by comparing these results with the manual annotations (when present)

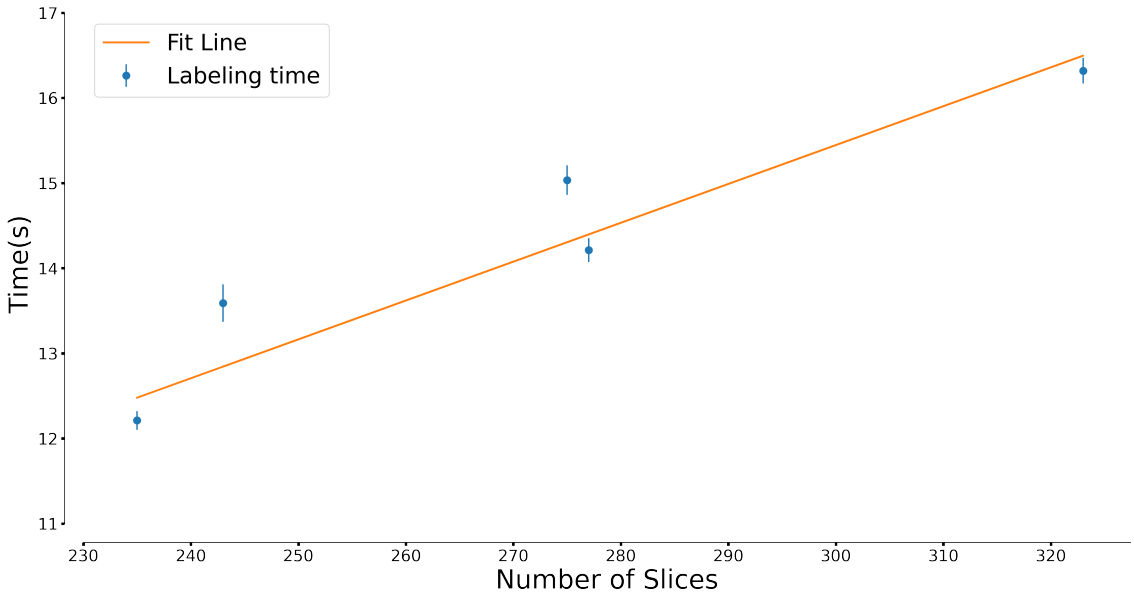



**Figure 3.2:** *Labelling time vs number of slice. This time incorporates also the creation of the multi-channel image. As we expected increase linearly with the number of slices and achieve a segmentation in less that* 17 *sec.*

To perform the segmentation I have estimated the set of centroids considering 10 CT scans from the available datasets, carefully selected to achieve a balanced representation for each cluster. The segmentation was performed using as hardware the servers of the Department of Physics and Astronomy(DIFA), and the segmentation of each patient have taken less than 3 minutes.

In Figure 3.3 I have reported a comparison between the achieved segmentation and the manual annotation as well as the original image. As we can see the lesion areas, seem correctly identified. The resulting segmentation and the annotation seem to be agreement. We can observe that the in, the first case, the annotation find some spots outside the main GGO area.

In Figure 3.4 I have provide a zoom on these interesting areas. We can see that the pipeline segmentation well identified the main opacities. We can spot that the annotation tends to consider as GGO a larger region than the pipeline. We can also notice that in the provided annotation also some regions that seem to be healthy are segmented.

In Figure 3.5 we can see the 3D structure of the lung regions with the identified GGO and CS. In green, we can observe the pipeline results; in pink the manual annotation. As we can see the two regions seem to be consistent. Again we can spot that the annotation tends to consider a larger area than the pipeline.

I also have matched results and annotation in a quantitative way. Using the 5 ground truth (accurate and validated manual segmentations) I have matched pipeline and annotation considering sensitivity and specificity. Moreover, in collaboration with the Department of Diagnostic and Preventive Medicine of the Policlinico Sant'Orsola - Malpighi, that the segmentation was submitted to five experts in order to make a blind evaluation, comparing the pipeline segmentation with annotation.

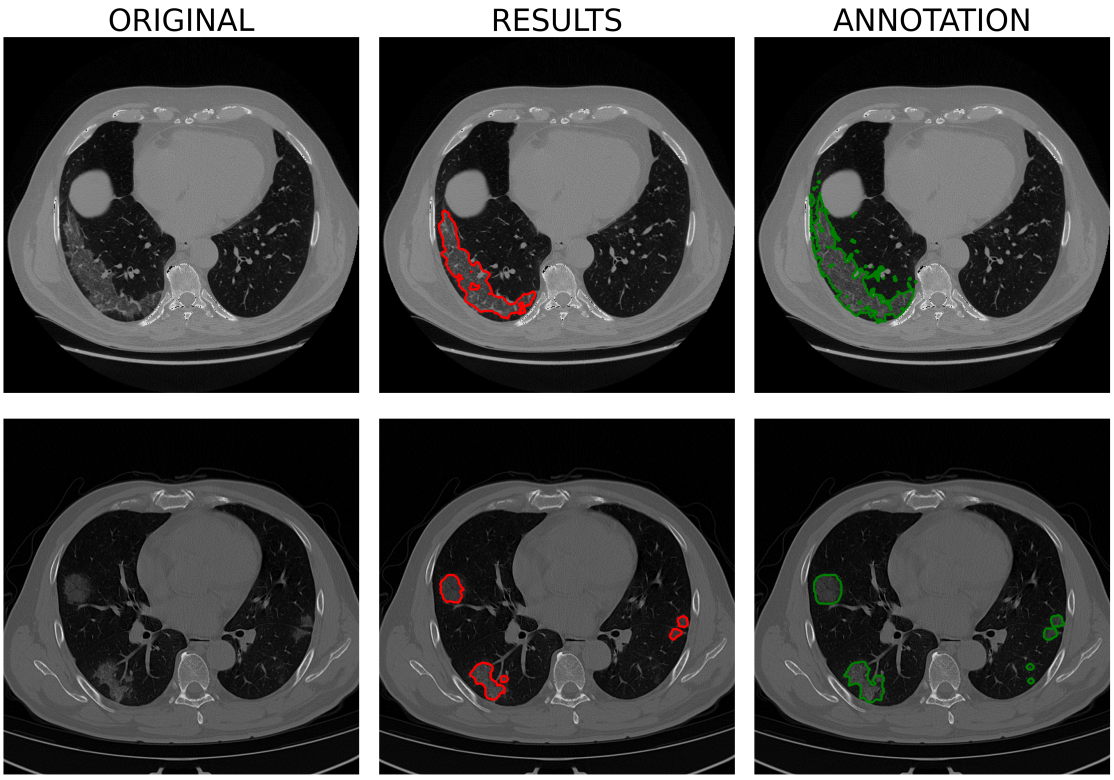



**Figure 3.3:** *Comparison between the achieved segmentation (red) and the annotation (green). We clearly see that the GGO and CS areas are well identified and segmented.*

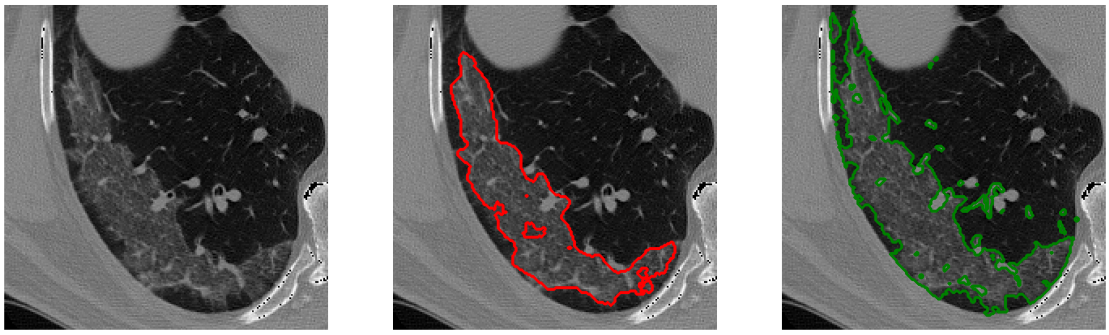


**Figure 3.4:** *Focus on the lesion area. From left to right we can observe the original image, the predicted GGO areas and the manual annotation. We can observe that the annotation identify a larger area and also some spots outside the lesions which seems to be healthy tissue.*

### 3.3.1 Comparison with Manual Annotations

To check the pipeline performances, I have compared the obtained segmentation with the manual annotation. To do that I have considered 5 scans from the Sant Orsola dataset for which was available also a ground truth. This ground truth consists in a semi-automatic segmentation made with a certified software and refined by an expert with more than 5 years of experience. After that, the segmentation was validated by 5 experts with at least 2 year of experience. This segmentation process takes several days.

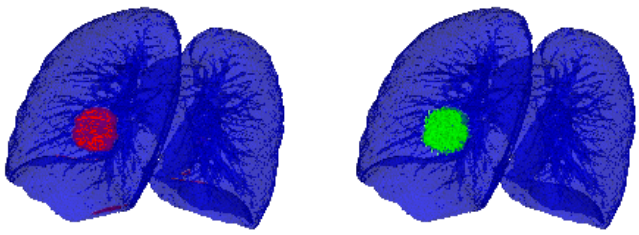


**Figure 3.5:** *3D representation of lung (blue) and identified GGO and CS. From left to right the pipeline segmentation (red) and the manual annotation (green). We can see that the two segmentation seems to agree. The area detected by the pipeline seems to be lower than the manual segmentation.*

To compare annotation and pipeline segmentation, I have computed the *sensitivity* and *specificity*:

**Sensitivity** refers to the ability to correctly detect ill areas. It is defined as the total number of voxels correctly classified as opacities (True Positives), over the total number of positives (True Positives + False Negatives) :

$$Sensitivity = \frac{TruePositive}{TruePositive + FalseNegatives} \tag{3.1}$$

**Specificity** relates to the ability to correctly reject healthy areas. Is defined as the number of rejected pixels(True Negative) against the total number of healthy areas (True Negative + False Positives) :

$$Specificity = \frac{TrueNegative}{TrueNegative + FalsePositives} \tag{3.2}$$

The results are displayed in Table 3.1.

| | Predicted | | Annotation | |
|---|---|---|---|---|
| | Sensitivity | Specificity | Sensitivity | Specificity |
| Patient 1 | 0.412 | $\sim 1.00$ | 0.676 | 0.999 |
| Patient 2 | 0.399 | $\sim 1.00$ | 0.698 | 0.995 |
| Patient 3 | 0.570 | $\sim 1.00$ | 0.653 | 0.999 |
| Patient 4 | 0.512 | $\sim 1.00$ | 0.325 | 0.999 |
| Patient 5 | 0.628 | $\sim 1.00$ | 0.974 | 0.999 |

**Table 3.1:** *Sensitivity and Specificity for the pipeline segmentation and annotation. As a ground truth was used a sem-automatic segmentation made and evaluated by* 5 *experts with at least* 2 *years of experience.*

The first thing we can notice is that both the method have a high specificity. That means that they rarely give positives results for healthy regions (lower type I error rate): there is a low probability to obtain false positives. The situation changes

when we consider specificity. We can see that the annotation has achieved a better sensitivity than the pipeline. That means that they rarely give negative results for GGO regions (lower type II error rate). However, this coefficient does not take into account the rate of false positives, that means we cannot ensure that the detected GGO areas are really sick.

This in agreement to the fact that, generally, the operator tends to include large lesion areas. However, that is not always the case, since will depends on the operator that performs the segmentation (subjectivity of these techniques).

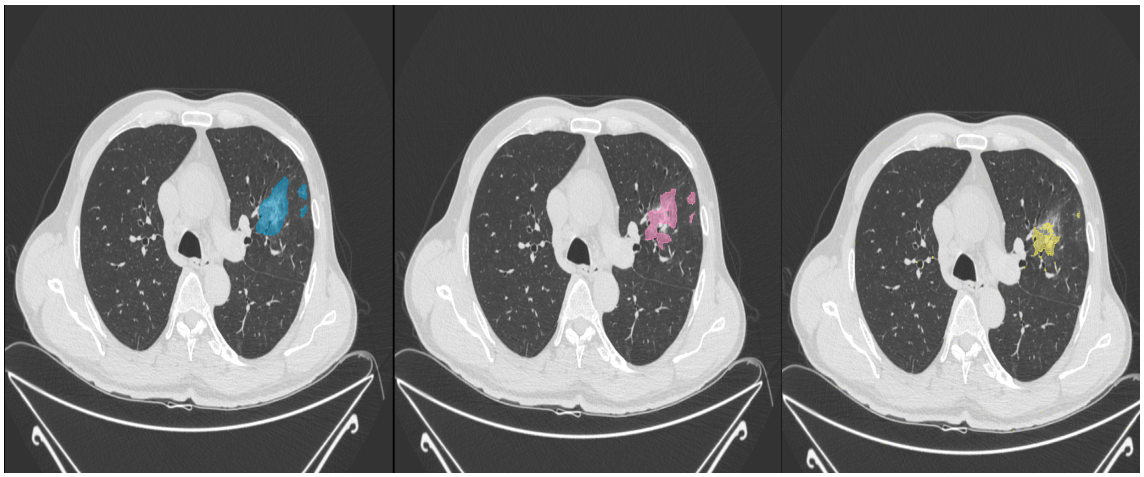


**Figure 3.6:** *Comparison between the ground truth (blue), the pipeline segmentation (pink) and the manual annotation (yellow). We can see that the segmentation obtained by the automatic pipeline is better than the one of the manual annotation.*

In Figure 3.6 I have reported the results of the segmentation of Patient 4. We can see the ground truth (blue), the pipeline segmentation (pink) and the annotation (yellow). In this case, we can see that both the pipeline and the annotation correctly identified the lesion areas. However, we can see that the annotation is missing a lot of lesions. On the other hand, the segmentation achieved by the pipeline seems to correctly segment the whole areas.

In Figure 3.7 vI have reported the segmentation results for the third patient. We can see the ground truth (blue), the pipeline segmentation (pink) and the annotation (yellow). Also, in this case, the pipeline seems to correctly identify the opacity. In this case, also, the annotation seems to agree with the ground truth.

Up to now, I have considered only two cases in which the GGO and CS regions are well defined with high contrast respect to healthy lung volume. In Figure 3.8 I have reported axial, sagittal and coronal view of the ground truth (blue) and pipeline segmentation (pink) for the first patient. This patient presents a low volume of GGO and CS. Moreover, the identification is difficult due to the low contrast between lesion areas healthy lung volume. As we can see the lesion areas are correctly identified even if some misclassified regions are presents.

In the end, I have to point out that the annotation and ground truth since are obtained by the semi-automatic method, requires trained personnel and several hours (the first) or days (the seconds). Moreover, the automatic method has the advantages of the speed and independence from an external operator.

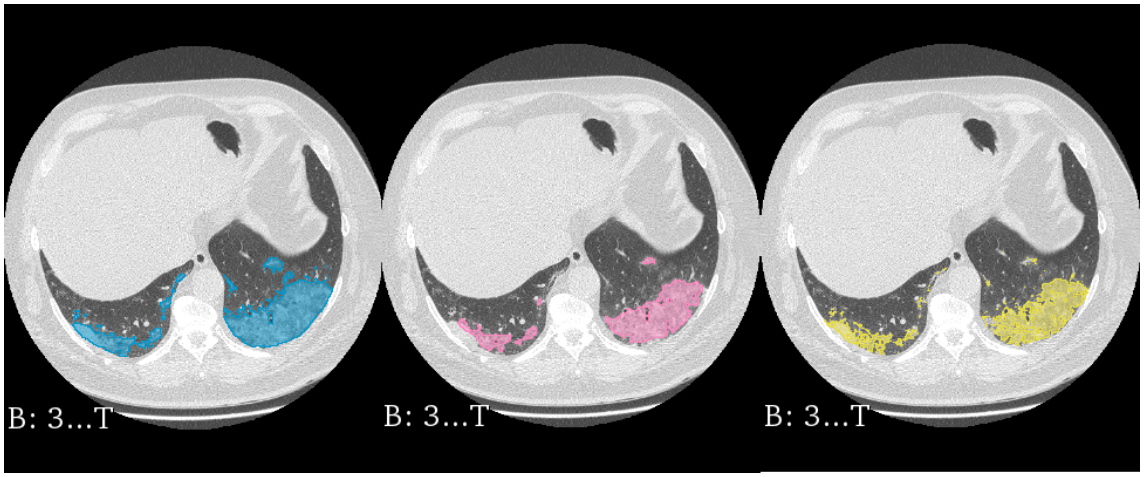



**Figure 3.7:** *Comparison between the ground truth (blue), the pipeline segmentation (pink) and the manual annotation (yellow). We can see that the GGO an CS areas are correctly identified.*

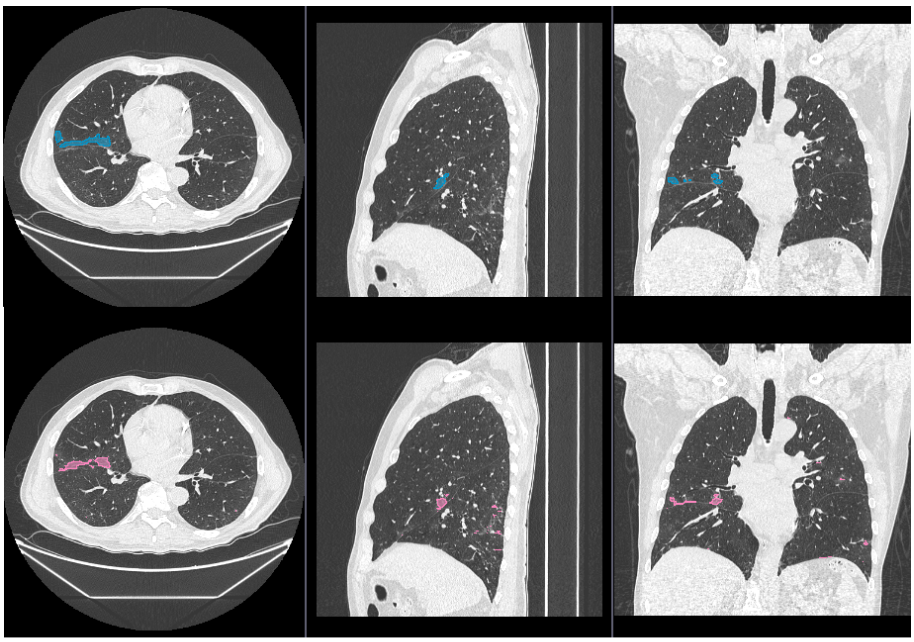


**Figure 3.8:** *Comparison between the gold standard segmentation(blue) and the pipeline results(pink) for an axial, sagittal and coronal view of a patient with a low involvement of lung volume. We can see that the main lesion areas are identified, even if an underestimation of the total volume is present together with some small misclassified points.*

### 3.3.2 Expert Evaluation

Another measure for the pipeline performances was a blind evaluation. In collaboration with the Department of Diagnostic and Preventive Medicine of the Policlinico Sant'Orsola - Malpighi, 3 experts with at least 2 years of experience have compared the pipeline segmentation and the reference labels (Annotation) provided within the dataset (obtained by semi-automatic technique).

To perform the evaluation, I have randomly selected 10 patients from the three datasets and organized the scans as follows: For each patient I have displayed, slice by slice, two images: the scans with the pipeline segmentation (the one to test), and the semi-automatic labels provided within the dataset (control) as in. Figure 3.9.

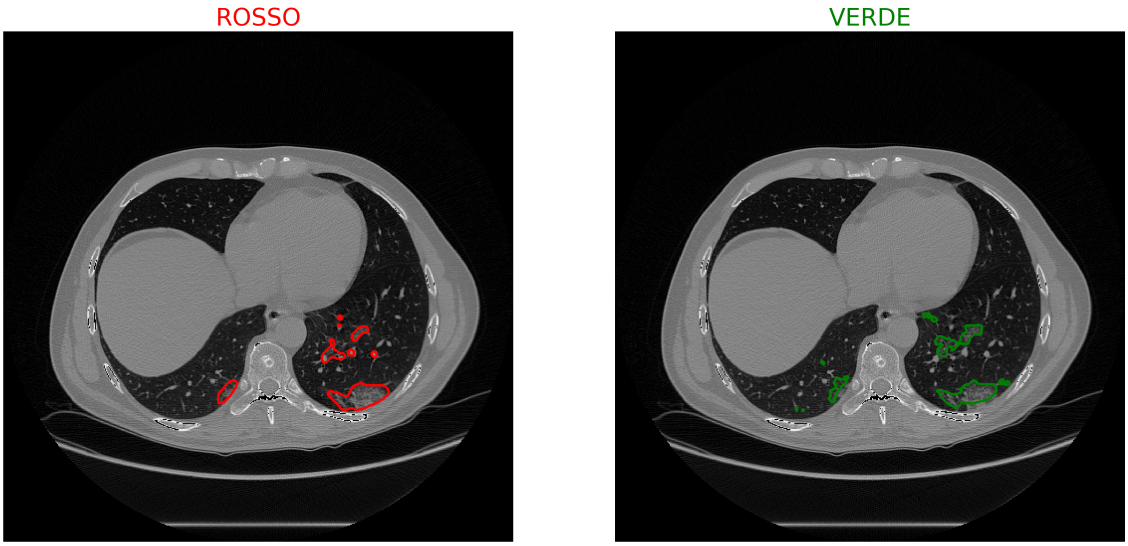



**Figure 3.9:** *Example of comparison submitted to the experts. The two images report the segmentation of the same slices achieved with two different techniques: Manual and pipeline. Was asked to the expert to decide which one is better, without knowing which technique is used to obtain the segmentation.*

The scans, organized in this way, was submitted to experts, who have been asked to decide for each scan which segmentation is better or if the quality is equal. For each sample, the whole scan was submitted, considering also the slices without lesion.

The core of this method is that the experts do not know which scan corresponds to a specific segmentation technique, so this will lead to an ideally unbiased evaluation. To ensure that, the order of segmentation was shuffled between patients. The expert has validated the scans independently, so for each patient, I have at most 3 different results.

| Patient | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Pipeline | 0.06 | 0.28 | 0.06 | 0.92 | 0.06 | 0.41 | 0.13 | 0.04 | 0 | 0.35 |
| Annotation | 0.60 | 0.23 | 0.80 | 0.00 | 0.51 | 0.27 | 0.53 | 0.01 | 0.43 | 0.13 |
| None | 0.34 | 0.49 | 0.14 | 0.08 | 0.42 | 0.32 | 0.34 | 0.95 | 0.57 | 0.52 |

**Table 3.2:** *Rate of positive evaluation for each class.*

In Table 3.2 I have reported the results of this evaluation. For each patient I have displayed the rate of positive evaluation for the pipeline segmentation and the annotation. Moreover I have also reported the rate of equal evaluation (None). Notice that this analysis takes into account also the slices in which there aren't lesions. In this way also, the false positives were takes into account.

As we can see the pipeline seems to return a better segmentation in more or less a half of the total cases. We have to point out that a segmentation does not mean

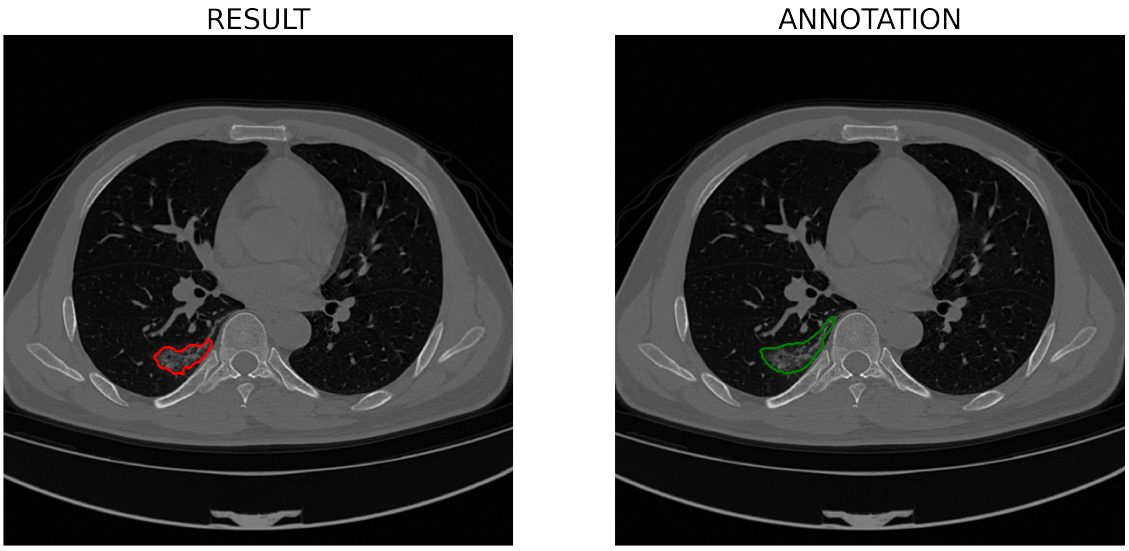



**Figure 3.10:** *Pipeline results vs Manual annotation for the $9_{th}$ patient. As we can see, even if the manual segmentation were classified abetter thatn the pipline one, the last seems to be consistent.*

the achievement of an inconsistent segmentation, but only that one segmentation is more accurate.

In Figure 3.10 I have reported a matching between the pipeline segmentation (red) and the manual annotation. As we can see the two results are consistent. However, the pipeline tends to underestimate the lesion area.

In Figure 3.11 I have matching the segmentation results for patient 10. The pipeline (red) has evaluated to provide a better lesion identification than the annotation (green). We can notice that in this case some healthy regions are misclassified by the operator, that are correctly removed by the pipeline.

Moreover, we can point out that, at least in the most typical cases, the difference between the pipeline segmentation and annotation is due mainly on the second one, since the subjectivity to the operator experience and the available segmentation time (hour or days) affect the quality. On the other hand, the developed approach does not suffer from these drawbacks, since it is automatic and fast.

| Class | Score rate |
|---|---|
| Pipeline | 0.31 |
| Annotation | 0.33 |
| None | 0.35 |

**Table 3.3:** *Score rate of positive evaluation for each class. As we can see the results seems to be uniformly distributed.*

In the end, as a final consideration, I have measured the rate of positive evaluation for each class, considering all the results provided. The results are reported in Table 3.3 from we can notice that the pipeline and annotation give positives results more or less with the same rate. Given two different labels it usually it is difficult to spot which is obtained by the pipeline and which belong to an annotation.

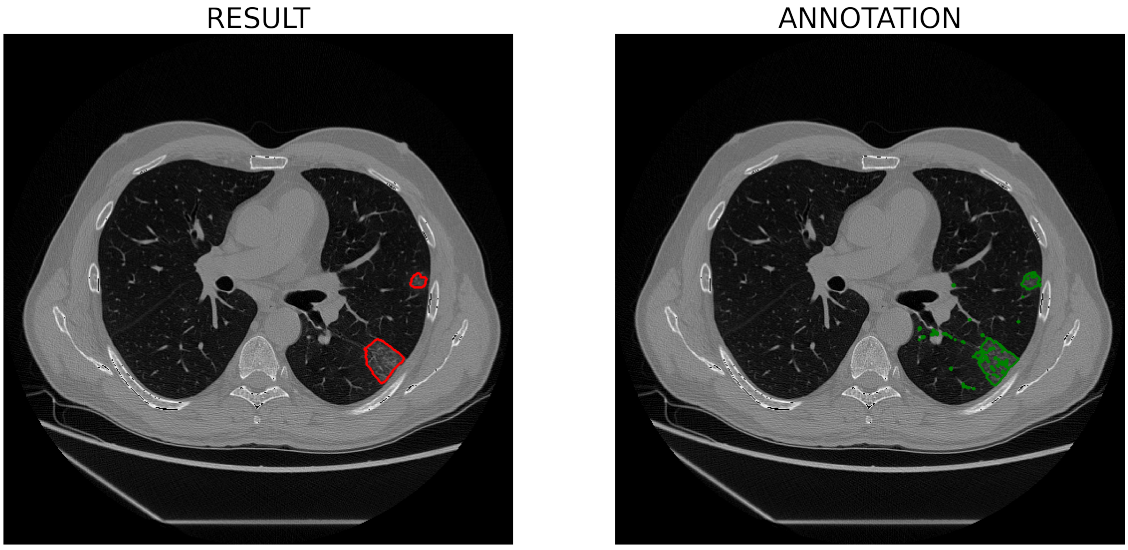



**Figure 3.11:** *Pipeline results vs Manual annotation for the* $10_{th}$ *patient. As we can see, even if the manual segmentation were classified abetter thatn the pipline one, the last seems to be consistent. We can notice that the oparator has selected also healthy tissues as GGO.*

# Conclusions

In this work of thesis, I have developed, implemented and tested an automated pipeline for the identification of Ground Glass Opacities and Consolidation in chest CT scans of patients affected by COVID-19.

As a preliminary step, I have performed a lung segmentation by using a pre-trained U-Net. This step is followed by a bronchial removal, to reduce the number of false positives. To perform the GGO and CS segmentation, I have applied the colour quantization. It allows identifying the different areas inside the lung, grouping them by colour similarity. Since the lesions involve many closest voxels, I have used the multi-channel properties of digital images to encode also neighbouring information, this was done by assigning at each channel a different function of the image (median blurring, gamma correction, neighbourhood standard deviation and CLAHE). To achieve the segmentation, I have found the characteristic colour of each tissue, performing a k-means clustering in the colour space. This set can be used to segment several scans by assigning each voxel to the nearest colour.

The pipeline was tested of 3 datasets by comparing the segmentation with manual annotation, using specificity, sensitivity and a blind evaluation made by experts.

The pipeline has shown to achieve segmentation consistent with the annotation in a small amount of time (less than $3\,min$) and does not require the interaction with trained personnel. The segmentation presents a high specificity with a small rate of false positives. Even if the lesion areas are underestimated, the expert evaluation has shown that better segmentation were achieved in the 31% of the slices by the pipeline and for the 33% by the annotation. In the remaining 35% equal performances were detected. However, in any case, the segmentation and annotation seem to be consistent.

Further developing is possible, like embedding of spatial information (not considered in this work), fine training and sensitivity. In the end, after these preliminary tests, the colour quantization has shown to be a suitable approach to face this kind of problems.

# Bibliography


[1] Zhao Fu and et al. "CT features of COVID-19 patients with two consecutive negative RT-PCR tests after treatment." In: *Scientific reports* (July 2020). DOI: 10.1038/s41598-020-68509-x.

[2] Huang C, Wang Y, and et al. "Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China". In: *Lancet* (Feb. 2020). DOI: 10.1016/S0140-6736(20)30183-5.

[3] Adam Bernheim et al. "Chest CT Findings in Coronavirus Disease-19 (COVID-19): Relationship to Duration of Infection". In: *Radiology* 295.3 (2020). PMID: 32077789, p. 200463. DOI: 10.1148/radiol.2020200463. eprint: https://doi.org/10.1148/radiol.2020200463. URL: https://doi.org/10.1148/radiol.2020200463.

[4] Tao Ai et al. "Correlation of Chest CT and RT-PCR Testing for Coronavirus Disease 2019 (COVID-19) in China: A Report of 1014 Cases". In: *Radiology* 296.2 (2020). PMID: 32101510, E32–E40. DOI: 10.1148/radiol.2020200642. eprint: https://doi.org/10.1148/radiol.2020200642. URL: https://doi.org/10.1148/radiol.2020200642.

[5] Collins J Stern E J. "Ground-glass opacity at CT: the ABCs". In: *American Journal of Roentgenology* 169 (1997), pp. 355–366. DOI: doi:10.2214/ajr.169.2.9242736.

[6] Ma Jun et al. *COVID-19 CT Lung and Infection Segmentation Dataset*. Version Verson 1.0. Zenodo, Apr. 2020. DOI: 10.5281/zenodo.3757476. URL: https://doi.org/10.5281/zenodo.3757476.

[7] N.A. Vladzymyrskyy A.V. Ledikhova N.V. Gombolevskiy V.A. Blokhin I.A. Gelezhe P.B. Gonchar A.V. Morozov S.P. Andreychenko A.E. Pavlov and Chernina V.Y. *MosMedData: Chest CT Scans With COVID-19 Related Findings*. Version Verson 1.0. 2020. URL: https://mosmed.ai/.

[8] Dzung L. Pham, Chenyang Xu, and Jerry L. Prince. "Current Methods in Medical Image Segmentation". In: *Annual Review of Biomedical Engineering* 2.1 (2000). PMID: 11701515, pp. 315–337. DOI: 10.1146/annurev.bioeng.2.1.315. eprint: https://doi.org/10.1146/annurev.bioeng.2.1.315. URL: https://doi.org/10.1146/annurev.bioeng.2.1.315.

[9] D. Withey and Z. J. Koles. "A Review of Medical Image Segmentation: Methods and Available Software". In: 2008.

[10] Dr J P Chaudhari Pooja V. Supe Prof. K. S. Bhagat. "Image Segmentation and Classification for Medical Image Processing". In: *International Journal on Future Revolution in Computer Science & Communication Engineering* 5.1 (), pp. 45–52.

[11] Michele Larobina and Loredana Murino. "Medical Image File Formats". In: *Journal of Digital Imaging* 27 (2 2014). ISSN: 27. DOI: 10.1007/s10278-013-9657-9.

[12] David Arthur and Sergei Vassilvitskii. "K-Means++: The Advantages of Careful Seeding". In: vol. 8. Jan. 2007, pp. 1027–1035. DOI: 10.1145/1283383.1283494.

[13] Laurence Morissette and Sylvain Chartier. "The k-means clustering technique: General considerations and implementation in Mathematica". In: *Tutorials in Quantitative Methods for Psychology* 9 (Feb. 2013), pp. 15–24. DOI: 10.20982/tqmp.09.1.p015.

[14] Ozturk Celal, Hancer Emrah, and Karaboga Dervis. "Color Image Quantization: A Short Review and an Application with Artificial Bee Colony Algorithm". In: *Informatica* 25.3 (2014), pp. 485–503. ISSN: 0868-4952. DOI: 10.15388/Informatica.2014.25.

[15] Johannes Prayer Forian Pan Jeanny Röhrich Sebastian Prosch Helmut Langs Georg Hofmanninger. "Automatic lung segmentation in routine imaging is primarily a data diversity problem, not a methodology problem". In: *European Radiology Experimental* 4 (2020). ISSN: 1. DOI: 10.1186/s41747-020-00173-2. URL: https://doi.org/10.1186/s41747-020-00173-2.

[16] Nico Curti. *Implementation and optimization of algorithms in Biomedical Big Data Analytics*. https://nico-curti2.gitbook.io/phd-thesis/. 2019.

[17] J. Austin et al. "Glossary of terms for CT of the lungs: Recommendations of the Nomenclature Committee of the Fleischner Society". In: *Radiology* 200 (Sept. 1996), pp. 327–31. DOI: 10.1148/radiology.200.2.8685321.

[18] Hoa Nguyen Johannes Hofmanninger. *Automated lung segmentation in CT under presence of severe pathologies*. https://github.com/JoHof/lungmask. 2020.

[19] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[20] Bradley Lowekamp et al. "The Design of SimpleITK". In: *Frontiers in Neuroinformatics* 7 (2013), p. 45. ISSN: 1662-5196. DOI: 10.3389/fninf.2013.00045. URL: https://www.frontiersin.org/article/10.3389/fninf.2013.00045.

[21] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[22] Biondi Riccardo et al. *COVID-19 Lung Segmentation*. https://github.com/RiccardoBiondi/segmentation. 2020.

[23] Tim Head et al. *scikit-optimize/scikit-optimize*. Version v0.8.1. Sept. 2020. DOI: 10.5281/zenodo.4014775.