Graduation Project Report

to obtain:

The National Engineering Diploma from Tunisia Polytechnic School

Deep Learning for medical imaging

Lung cancer candidates generation

Presented By/Elaborated by:

Marwen KRAIEM

Within/In:

Université de Moncton

Supported 01/10/2019

Mr	First name& NAME	grade - Institution	President
Mr	Moulay AKHLOUFI	Professor – Université de Moncton	Supervisor
Mr	Rabbeh Attia	Professor – Ecole Polytechnique de Tunisie	Academic Supervisor
Mr	First name& NAME	grade - Institution	Guest

Academic Year: 2018 - 2019

″The good-news stories in medicine are early detection,
early intervention"

Thomas R. Insel

Acknowledgements

*F*irstly, I would like to express my sincere gratitude to my supervisors Prof. Moulay AKHLOUFI and Prof. Rabbah ATTIA, who extended their support and helped me to achieve my best.

*B*ig thanks goes to Andy Couturier who has provided all the support in the implementation and valuable advices and comments on my work.

*A*lso, I would like to thank all PRIMe's members for the warm welcome and the great support.

*F*inally, I must express my very profound gratitude to my family and freinds for providing me with unfailing support and continuous encouragement throughout my years of study. This accomplishment would not have been possible without them.

Thank you.

Abstract

Lung cancer is one of the most common cancer in the world. Lots of researchs were conducted to help radiologists detect him in the early stage. A computer aided diagnosis could be a good solution with the development of new technologies such us deep learning. In this context, we propose a two-stage nodule detection framework that detects nodule candidate with convolution neural networks trained on 3D volume, followed by 3D neural network architecture to reduce false detection.

We present in this report the first part of a computer aided system for detecting cancer from computed tomography scans. The implemented model generates nodules candidates using a deep learning approach to be classified by the second part of the proposed framework. We tried to use a modified version of 3D-UNet to segment nodules and use blob detection algorithm to generate coordinates and diameters of the segmented nodules from LUNA16 dataset. The results found seems to be interesting since we achieved a sensitivity of 92.1 % and generated an acceptable number of candidates.

Keywords: Lung cancer, LUNA16, Deep Learning, Convlutional Neural Network, medical imaging.

Résumé

Le cancer du poumon est l'un des cancers les plus courants dans le monde. Des nombreuses recherches ont été menées pour aider les radiologues à le détecter à un stade précoce. Un diagnostic assisté par ordinateur pourrait être une bonne solution avec le développement de nouvelles technologies telles que l'apprentissage profond. Dans ce contexte, nous proposons un framework de détection de nodules en deux étapes qui détecte les candidats nodules avec des réseaux de neurones convolutionnels formés sur le volume 3D en premier lieu, suivis d'une architecture de réseau neuronal 3D pour réduire les fausses détections.

Nous présentons dans ce rapport la première partie de ce système assisté par ordinateur pour la détection du cancer. Le modèle mis en œuvre génère des candidats en utilisant une approche d'apprentissage profond. Ces candidats seront classés dans la deuxième partie du framework proposé. Nous avons essayé d'utiliser une version modifiée de 3D-UNet pour segmenter les nodules et d'utiliser un algorithme de détection de blob pour générer les coordonnées et les diamètres des nodules segmentés. Les résultats trouvés semblent intéressants puisque nous avons atteint une sensibilité de 92,1% et généré un nombre acceptable de candidats.

Mots-clés : Cancer du poumon, LUNA16, apprentissage profond, réseau neuronal convolutionnel, imagerie médicale.

Contents

Acknowledgements **i**

Abstract **i**

Résumé **i**

List of Figures **iv**

List of Tables **vi**

Introduction **1**

1 General framework of the project **3**

1.1 The host organization 3

1.1.1 Université de Moncton 3

1.1.2 PRIMe group 3

1.2 Problem formulation 4

1.2.1 Main goal 4

1.2.2 The problem of the internship 4

2 Fundamentals **6**

2.1 Medical imaging 6

2.2 Image segmentation 8

2.2.1 Semantic segmentation and Instance segmentation 8

2.2.2 Segmentation approaches 9

2.2.2.1 Thresholding 9

2.2.2.2 Region based segmentation 10

2.2.2.3 Edge based segmentation 11

2.2.2.4 ANN segmentation 11

2.3 Deep learning 11

2.3.1 Artifitial neural networks 11

2.3.1.1 Introduce ANN . . . 11
2.3.1.2 Training an ANN . . . 13
2.3.2 Convolutional Neural Network (CNN) . . . 14
2.3.2.1 CNN layers . . . 14
2.3.2.2 Filter hyperparameters . . . 15
2.3.3 Regularization for Deep Learning . . . 17
2.3.3.1 Motivation . . . 17
2.3.3.2 Dropout . . . 17
2.3.3.3 Batch Normalization . . . 18
2.3.3.4 Early stopping . . . 19
2.3.3.5 Data augumentation . . . 19
2.4 Blob detection . . . 20
2.4.1 The Laplacien of Gaussian (LoG) . . . 20
2.4.2 The Difference of Gaussian (DoG) . . . 20

3 Proposed approach 22
3.1 The general approach . . . 22
3.2 Dataset . . . 23
3.2.1 CT scans . . . 23
3.2.2 Annotations . . . 24
3.3 Preprocessing . . . 24
3.3.1 Motivation . . . 24
3.3.2 Converting to HU . . . 24
3.3.3 Resampling . . . 25
3.3.4 Normalization . . . 26
3.3.5 Patches Extraction . . . 26
3.4 Lung segmentation . . . 27
3.5 Nodules segmentation . . . 29
3.5.1 Motivation . . . 29
3.5.2 Neural Network architecture . . . 30
3.5.3 Metrics and loss function . . . 31
3.6 Exctract Nodules centroides . . . 31

4 Experimental results 33
4.1 Tools . . . 33
4.2 Model training . . . 34
4.2.1 Hardware charcteristics . . . 34
4.2.2 Training . . . 34

4.3 Regularization 35
4.3.1 Data augumentation 35
4.3.2 Dropout and Batch normalization 35
4.4 Results and interpretations 35
4.5 Results comparaison 37

Conclusion and Future perspectives 38

Appendices 39

Bibliography 44

List of Figures

1 Cancer deaths by type in 2017 2

1.1 PRIMe’s logo 4

2.1 2D slice form CT scan containing a nodule 8
2.2 Example of instance and segmentation 9
2.3 Example of Thresholding application 10
2.4 splitting image into regions 10
2.5 Model of a neuron labeled k 12
2.6 An example of ANN 13
2.7 Image explaining 3D convolution 14
2.8 Difference between max pooling and average pooling 15
2.9 Filter’s parameters 16
2.10 Stride representation 16
2.11 Different padding modes 17
2.12 Neural network architecture before and after dropout 17
2.13 Early stopping process 19
2.14 Examples of data augumentation 19

3.1 Proposed Diagram for the given problem 22
3.2 Preprocessing Digram to extract the training patches 24
3.3 Histogram of all voxels in the CT scan 25
3.4 Resut of removing -2000 values 25
3.5 Example of an extracted patch and his ground truth 26
3.6 Step 1 27
3.7 Step 2 28
3.8 Step 3 28
3.9 Step 4 28
3.10 Step 5 29
3.11 Step 6 29
3.12 Network architecture 30

4.1 Curves of training the network on 40x40x40 patches 35
4.2 Displaying results 36

List of Tables

2.1 Hounsfield Unit scale 7

3.1 Information about the CT scan 23

4.1 Results of the trained network 36
4.2 Candidates centroids generation results 36
4.3 Sensitivity results 37
4.4 Results comparaison 37

Introduction

Lung Cancer is considered as the deadliest cancer in the world. Besides, according to the GCO (Global Cancer Observatory) the number of new lung cancer's cases in 2018 represents 11.6 % of all cancer types causing the death of 18.4 % [1] of the whole deaths caused by cancer.

This reason has encouraged many countries to invest money in developing early diagnosis of lung cancer. The NLST trial [2] showed that three annual screening with low-dose Computed Tomography (CT) reduces death rates considerably. This result conclude to the importance of radiologist's inspection of an overwhelming quantity of CT scan images. Since nodules detection is usually a difficult task to do, even by experienced doctors, the burden on radiologists increases enormously. With this expected increase in the number of preventive/early detection measures, researchers are concentrating in computerized solutions that alleviate the work of doctors, improve diagnostics's precision and speed up the analysis.

The detection of malignant nodules requests certain features to recognize and characterize these tumors. The probability that these nodules are carcinogenic is based on a sort of combination of some aspects and characteristics such as volume, shape, sphericity, solidity, etc. These features are used with some machine learning techniques to classify nodules to benign or malignant. The problem with machine learning frameworks is the necessity of many hand-crafted parameters to obtain better performance and reliable results.

The proposed solution is to build a CAD system using Deep Learning in our approach. Neural networks tend to give good results in extracting features and using them to make the good decision, after the training is done on huge amount of data.

In this work, we propose the first part of a CAD system for Lung cancer detection. This part is dedicated to detect possible nodules candidates to feed them to a classifier to choose between malgniant and bengniant nodules. The first part will introduce the host organization, define the main goal of the project and formulates the problem under study.

The second part of the report, will introduce the necessary theorical background and define some notions. The rest of the report is organized as follows: Chapter 3 then details the proposed solution and go through different steps done to arrive at the fixed goal. After that, chapter 4 illustrates the experimental results with proper discussion. Finally, concluding remarks and future research perspectives are drawn.

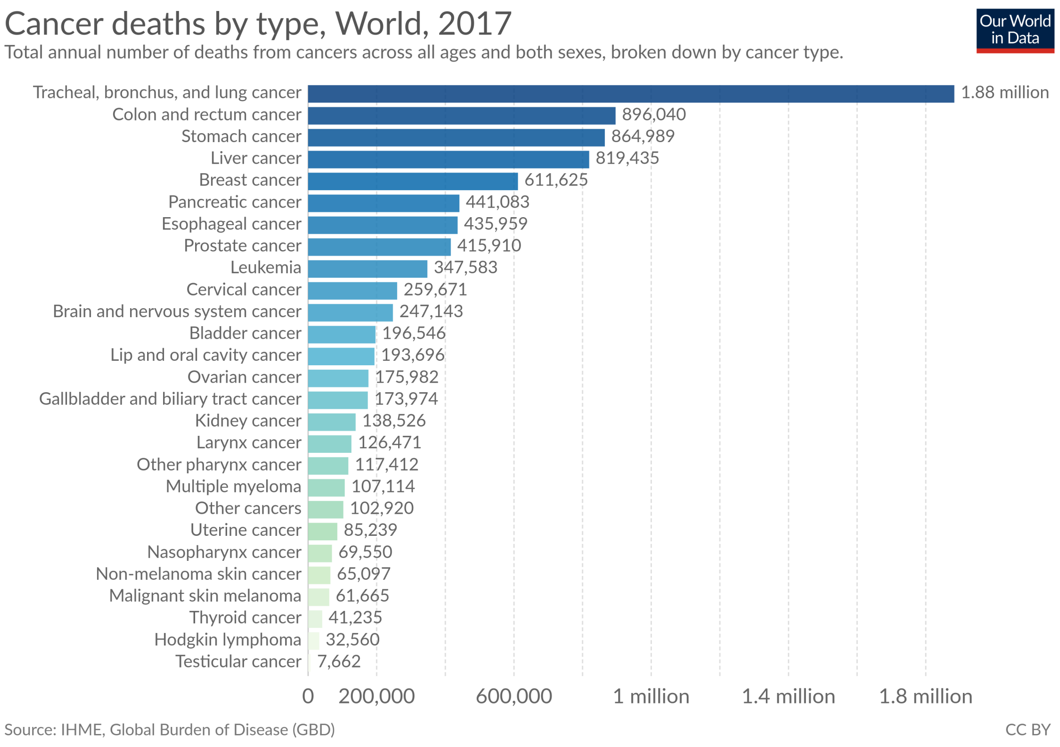


Figure 1 – Cancer deaths by type in 2017

Chapter 1

General framework of the project

Introduction

In this chapter, we start by presenting the host University and the research lab PRIMe. Then, we will define the problem and the main goal of the project.

1.1 The host organization

The internship is proposed by Mitacs to international student to realize a project in a canadian university. The host university is Université de moncton in PRIMe group.

1.1.1 Université de Moncton

The Université de Moncton is Canada's largest French-language university outside Québec. The university's three campuses are located in New Brunswick's francophone regions at Edmundston, Shippagan and Moncton [3]. The university offers 165 study programs, including 40 at the master's degree and doctoral level. Fields of study include administration, arts, education, engineering, forestry, law nursing, nutrition, psychology, sciences, social work, etc. It contains also more than 35 research centers, chairs and institutes. The university has a diversity in the student population with mix between local students from New Brunswick and students from francophone countries.

1.1.2 PRIMe group

The PRIMe (Perception, Robotics and Intelligent Machines) is a research group conducting research in Machine Learning, Deep Learning, Computer and Robot Vision, Drone Visio, Augumented reality and intelligent machines. The group is founded in 2017 by professor Moulay Akhloufi, to solve industrial problems in the areas of Defense and security, Medical diagnosis, Intelligent robotics, smart sensing, sport analytics and recommanders system.

Figure 1.1 – PRIMe's logo

1.2 Problem formulation

1.2.1 Main goal

The problem proposed to solve in this internship is a part of research project in lung cancer detection in an early stage. The work is based on a review made by another intern in the laboratory about the existing techniques in Lung cancer detection. The work proposed in this project is the study of the existing methods in lung cancer detection and the implementation of a powerful tool that detect cancer from given CT scans.

My supervisor proposed a pipeline with two stages:

— The first one is doing the semantic segmentation and the detection of nodules' candidates coordinates and diameters. This part tries to detect from the CT scan volume all the true nodules, but that usually includes a high number of false positives.

— The second part is the classification of these nodule to malignant and benignant to make the decision later on if the patient has or will have cancer in the future or not. The second step basically aims to reduce the large number of false positives generated on the previous step.

1.2.2 The problem of the internship

My work was concentrated on the first stage: we are going to use a 3D approach to detect nodules candidates to classify them in the next stage. The choice of the 3D method is made after seeing a review [4]. This review is realized by a chilien student in Prime Lab to explore the state-of-the-art of methods used to detect lung cancer from given CT scans. The review finished with the conclusion: Future works can further improve 3D Convolutional architectures for this purpose. Both the design of new architectures and the study of the existing ones could improve the performance and the computational cost of three-dimensional networks. So based on this conclusion, the work to be done is to explore more 3D Convolutional architectures and improve some existing system to detect nodules effiecently.

There are lots of obstacles to overcome to solve this part starting from the pre-processing, the optimization of number of operations, the limitation of the hardware capacity.

The most significant one is using the whole images as input due to:

1. The limited size of the GPU memory (6Gb), so that in order to fit the model into the GPU, the network needs to greatly reduce the number of features and/or the layers, which often leads to a significant drop in performance as the expressiveness of the network is much reduced.
2. The training time will be greatly prolonged since more voxels contribute to calculation of the gradients at each step and the number of steps cannot be proportionally reduced during optimization.
3. As the background voxels dominate the whole image, the class imbalance will cause the model to focus on background if trained with uniform loss, or prone to false positives if trained with weighted loss that favors the foreground voxels.

Conclusion

In this first chapter, we introduced the general context of the work, after having started with a presentation of the PRIMe research group and the context of the project. Then, we have described the main problem and we detailed our objectives.

Chapter 2

Fundamentals

Introduction

In this chapter, we will explain some theorical background and define some terms necessary to understand the work done in this internship. First, we define medical images and give the main difference between it and normal one. Then, we see necessary defenition about image segmentation. Finally, we finish with deep learning and tools used in this project.

2.1 Medical imaging

Imaging has an important role in the modern medicine. This technique is widely used by doctors to have a visual representations of the interior of a body to make then clinical analysis and choose then the best medical intervention.
There are several types of medical imaging:

1. Magnetic Resonance Imaging
2. Ultrasound
3. Nuclear Medicine Imaging
4. X-rays
5. Computed Tomography scan [Fig.2.1]

One of the most common medical images type is CT scanning, also known as computed axial tomography, or CAT scan is a type of medical technology that uses X-rays and computers to produce three-dimensional images of the human body. This type of imaging is essentially an extrapolation of the concept of an x-ray imaging which relies on the principle that an object will absorb or scatter x-rays of a particular energy in a manner dependent on its composition, quantified by the attenuation coefficient μ.

The attenuation coefficient μ of a substance is a function dependent on a variety of factors, but primarily reflective of the electron density of that substance. Therefore, denser substances and substances containing elements with many electrons will have higher attenuation coefficients [5].

In practice, Hounsfield units, attenuation coefficients normalized to the attenuation coefficient of water, are used in favor of attenuation coefficients. This is due to the fact that these units are suited to the examination of organisms primarily composed of water to highlight the body's soft tissues, including blood vessels. The Hounsfield unit of a tissue is defined by [5]:

$$H_{tissue} = \frac{\mu_{tissue} - \mu_{water}}{\mu_{water}} * 1000$$

Material	**HU**
Air	-1000
Lung tissue	-700 to -500
Fat	-120 to -90
Water	0
Kindey	20 to 45
Blood	13 to 50
Muscle	35 to 55
Liver	54 to 66
Soft tissue	100 to 300
Bones	300 to 1900

Table 2.1 – Hounsfield Unit scale

This table shows the HU scale of the most frequent materials inside the patient's body.

A CT scan is multi-perspective x-ray image. It rotates a point source of x-rays around a body to be imaged. Taking the calculations from a full rotation, it is possible to reconstruct the 2D slice of the object. Compilation of multiple slices allows for 3D reconstruction of the object [5].

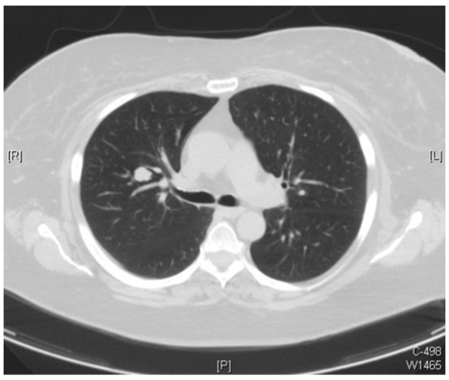


Figure 2.1 – 2D slice form CT scan containing a nodule

2.2 Image segmentation

Image segmentation is the process of partitioning a digital image into multiple segments. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.[6] Image segmentation is typically used to locate objects and boundaries in images. More precisely, the image segmentation is a sort of a classification problem where we assign a label to each pixel in an image such that pixels with the same label share certain characteristics.

The result of this operation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture.

2.2.1 Semantic segmentation and Instance segmentation

The goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented. Because we are predicting for every pixel in the image, this task is commonly referred to as dense prediction. This type of segmentation could be defined as a sort of partitioning an image to semantically meaningful parts and to classify each part as into one of the predetermined classes. So, semantic segmentation is basically understanding the role of each pixel in the image.

In the other hand, instance segmentation is one step ahead of semantic segmentation where in along with pixel level classification, we expect the computer to classify each instance of a class separately. This type of segmentation is a deeper version of semantic segmentation where we assign to pixel the object instance it belongs to.

Both types of segmentation have many sort of applications: atonomouns driving, remote sensing, industrial inspection, medical imaging analysis, etc.

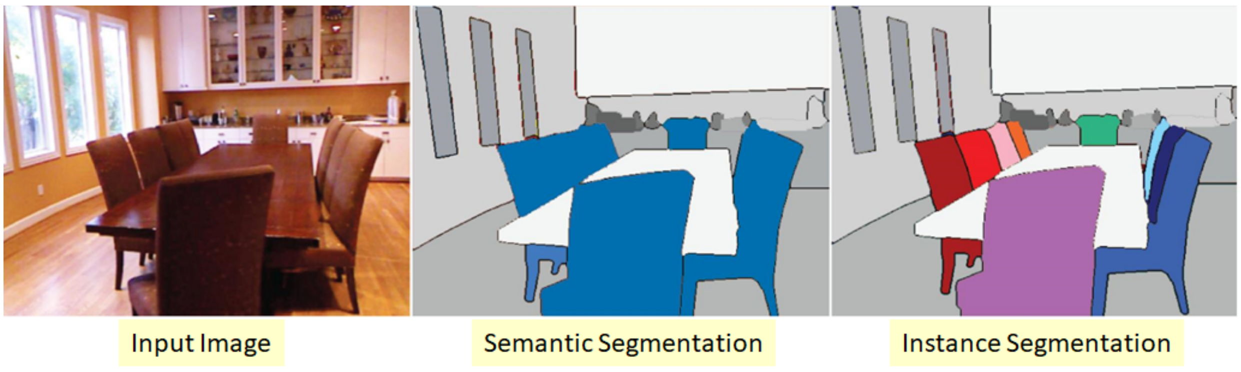


Figure 2.2 – Example of instance and segmentation
Source: https://i.stack.imgur.com/

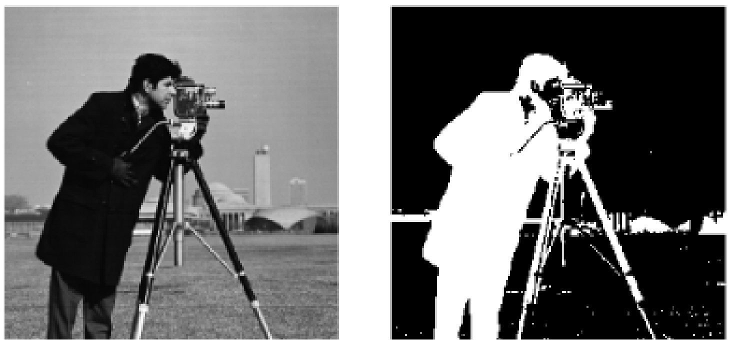
2.2.2 Segmentation approaches

In image segmentation there are sveral techniques used and each of these techniques has her own impotance. We will give some brief definition of the most commun technics in this field. Mathematical description is avoided for simplicity therefore all the techniques are described theoretically.

2.2.2.1 Thresholding

Thresholding methods are the simplest methods for image segmentation. These methods divide the image pixels (or voxels) with respect to their intensity level. These methods are used over images having lighter objects than background. The selection of these methods can be manual or automatic. In this method, we replace each pixel in an image with a black pixel/voxel if the intensity $I_{i,j}(orI_{i,j,k})$ is less than some fixed constant T, or a white pixel/voxel if the intensity is greater than this fixed T [7].(see figure 2.3)

$$\begin{cases} 0, & \text{if } I_{i,j} < T. \\ 1, & \text{otherwise.} \end{cases}$$



(a) Original image (b) Thresholded image

Figure 2.3 – Example of Thresholding application
Source: https://i.stack.imgur.com/

2.2.2.2 Region based segmentation

The region based segmentation methods are the methods that segments the image into many regions having similar characteristics. There are two basic techniques based on this method:

- **Region growing:** This method consist of dividing the image into various regions based on the growing of seeds (initial pixels). These seeds can be selected manually or automatically. Then the growing of seeds is controlled by connectivity between pixels and with the help of the prior knowledge of problem, this can be stopped [7].
- **Region splitting and merging:** This technique consist of splitting and merging for segmenting an image into different regions. Splitting stands for iteratively dividing an image into regions having similar characteristics and merging contributes to combining the adjacent similar regions [7].

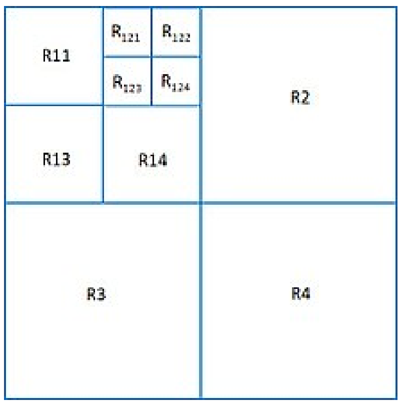


Figure 2.4 – splitting image into regions
Source: https:en.wikipedia.orgwikiSplit_and_merge_segmentation

2.2.2.3 Edge based segmentation

These methods are based on the rapid change of intensity value in an image. Edge detection techniques locate the edges where either the first derivative of the intensity is greater than a particular threshold or the second derivative has zero crossings. The first step to do in these methods is to detect the edges to connect them then together to form the object boundaries to segment the required regions. The basic two edge based segmentation methods are: Gray histograms and Gradient based methods.

To detect the edges one of the basic edge detection techniques like sobel operator, canny operator and Robert's operator etc can be used. Result of these methods is basically a binary image. These are the structural techniques based on discontinuity detection[7].

2.2.2.4 ANN segmentation

The artificial neural network based segmentation methods simulate the learning strategies of human brain for the purpose of decision making. Today, this method is mostly used for the segmentation of medical images. This modern method uses CNN (see section 2.3.2) to extract features and use them to make desicion on each pixel.

2.3 Deep learning

Deep learning, as a new area of machine learning research, is a process which allows the computer to learn to perform tasks which are natural for the brain like image recognition. Currently, deep learning (DL) methods have had a profound impact on computer vision and image analysis applications, such as image classification, segmentation, image completion and so on. Deep learning focuses on a specific category of machine learning called Artificial Neural Networks which is inspired by functionality of the human brain.

2.3.1 Artifitial neural networks

2.3.1.1 Introduce ANN

Like human brain composed of large number of neurons interconnected, ANN has an elementary unit called perceptron. The perceptron is a simplified artifitial model of the neuron [8]. In analogy to the mechanism of the real neuron, a perceptron takes a linear combination of its inputs followed by a nonlinear activation function φ as presented in figure 2.5.

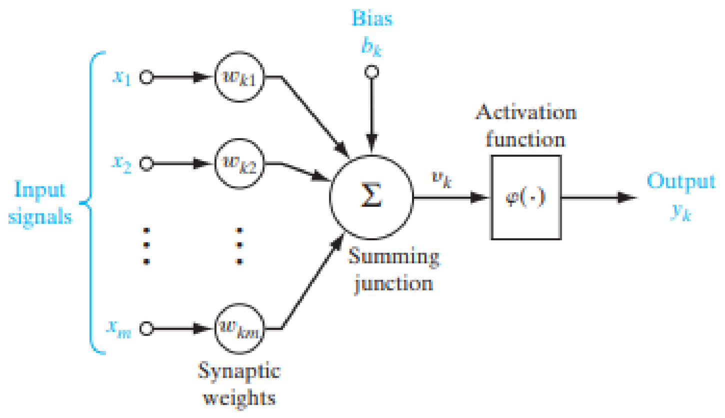


Figure 2.5 – Model of a neuron labeled k
Source: Neural networks and learning machines/Simon Haykin

The output of this perceptron could be expressed as:

$$y_k = \varphi(\sum_{i=1}^{N} w_i x_i + b_k)$$

where

— y_k is the output

— x_i are the inputs

— w_i are the weights

— b_k is the bias

— φ is the activation function

The activation function is generally chosen to be no linear such us *sigmoid* and *Rectified Linear Unit* (ReLU). (see appendix B)

The ANN mimic the human brain by hooking up many of this artificial neurons in a layer-by-layer structure where intermediate layers are called hidden layers.

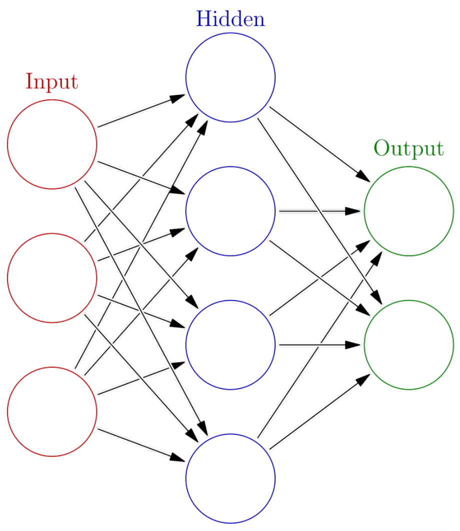


Figure 2.6 – An example of ANN

An artifitial neural network composed of three neurons as input, one hidden layer with five neurons and an output layer with two neurons.

2.3.1.2 Training an ANN

In order to set the optimal weights in an ANN network, a training process is needed. The most widely used approach for training neural networks is the supervised learning scheme by minimizing some loss error function **E** which is used to measure the distance between the predicted output of the network and the desired output . Several approach can be used to tackle this optimization problem such as genetic approaches or gradient based approaches. But the most widely used approach is the Stochastic Gradient Descent (**SGD**) method due to its simplicity and effectiveness in optimizing neural networks. For a set of training examples **x** with ground-truth labels **y**, the **SGD** method works as follows:

Algorithm 1 SGD Algorithm

```
Initialize all the network weights with random values
for epoch=1:n do
    for x=1:m do
        Compute the predicted output through a forward pass to the neural network.
        Compute the loss or error function E
        Compute the gradients of all weights by backpropagation of the output error to all the network parameters through a backward pass using the chain rule
        Update all the network weights by a specific amount in the negative direction of their corresponding gradients, according to the update rule $\Delta w_t^j = -\eta \frac{\delta E^t}{\delta w_t^j}$
    end for
end for
return weights
```

The update step of the weights is determined by the so-called learning rate Δ. This parameter is very critical in the training process [9] and it has crucial impact on the whole performance of the network. Recently, much research has been focused on proposing better approaches to integrate this learning rate in the learning process such as Adadelta, Adagrad and ADAM [9].

2.3.2 Convolutional Neural Network (CNN)

Proposed by Yann Lecun [10] is considered as the most used type of neural networks used in computer vision, image classification, object detection, etc. CNN is a class of deep neural networks, most commonly applied to images. These neural networks employ convolutions to extract features and use them to make predictions. The main componants of this type of neural network are layers.

2.3.2.1 CNN layers

— **Convolution layer:** Despite being a simple linear mathematical operation, Convolution is widely used in engineering and mathematics. The Convolution layer(or Conv layer) is the core building block of a Convolutional Network that does most of the computational heavy lifting. Most generally, we can think of a CNN as an artificial neural network that has some type of specialization for being able to pick out or detect patterns. This pattern detection is what makes CNNs so useful for image analysis [11].

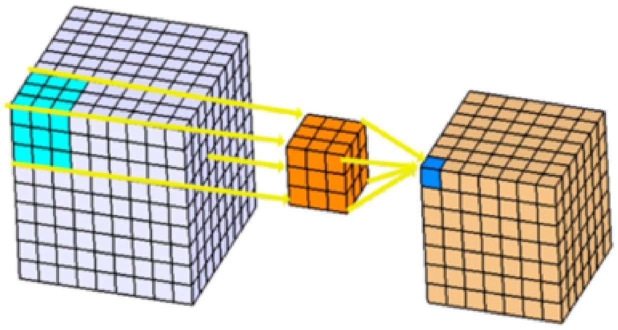


Figure 2.7 – Image explaining 3D convolution
Source: https://www.kaggle.com/shivamb/3d-convolutions-understanding-use-case

— **Pooling layer:** Pooling layers are used after CONV layers in the CNN to reduce the feature maps output. It does so by summarizing a spatial region of the feature map into one output value, this also reduces the number of required weights needed in next layers.
There are two main type of pooling layers:

- Maxpooling Layer: Calculate the maximum value for each patch of the feature map.
- Averagepooling layer: Calculate the average value for each patch on the feature map.

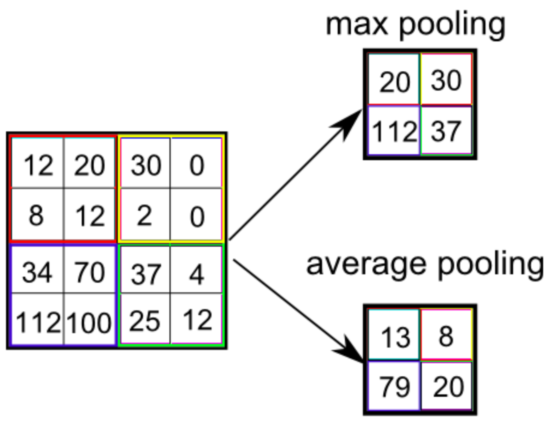


Figure 2.8 – Difference between max pooling and average pooling
Source: https://qph.fs.quoracdn.net/mainqimg-98ecf7ba49710bf56042d035a74505b6

— **Deconvolutional layer:** Deconvolution, also called transposed convolution in recent computer vision literature is another type of CNN layers. Transposed convolutions is used to get to the opposite direction of a normal convolution.[12].

— **Upsampling layer:** Upsampling layers are the symmetric operation of pooling or downsampling. As pooling operations are irreversible, producing the exact original feature map is impossible. Upsampling layer usually just take a one value input and repeat it over the 2D or the 3D local region.

— **Fully connected layer:** This layer is generally used to make the hight-level reasoning by connecting all the activations from the previous layer. it is usually used to perform classification based on features extracted by the previous layer.

2.3.2.2 Filter hyperparameters

Convolutional layers contain filters defined by some hyperparameters.

— **Dimensions:** The convolutional layer's parameters consist of a set of filters also known as kernels. These filters are designed to extract and detect some characteristics from the input. To If we have K filters of size F^d applied to an input containing C channels is an $F^d \times C$ volume that performs convolutions on the input of size $I_1 \times I_2 \times \ldots \times I_d \times C$ and produces an output feature map of size $O_1 \times O_2 \times \ldots \times O_d \times K$.

Where d is the dimension of our data, i.e d=2 if we have 2D data. F is the length of the filter, ,I_i is the length of the input in the dimension i and O_i is the length of the output in the dimension i defined as follow:

$$O_i = \frac{I_i - F + P_{end} + P_{start}}{S} + 1$$

Where S is the stride and P_{end} and P_{start} are the zero padding added respectivly at the end and at the start of the input.

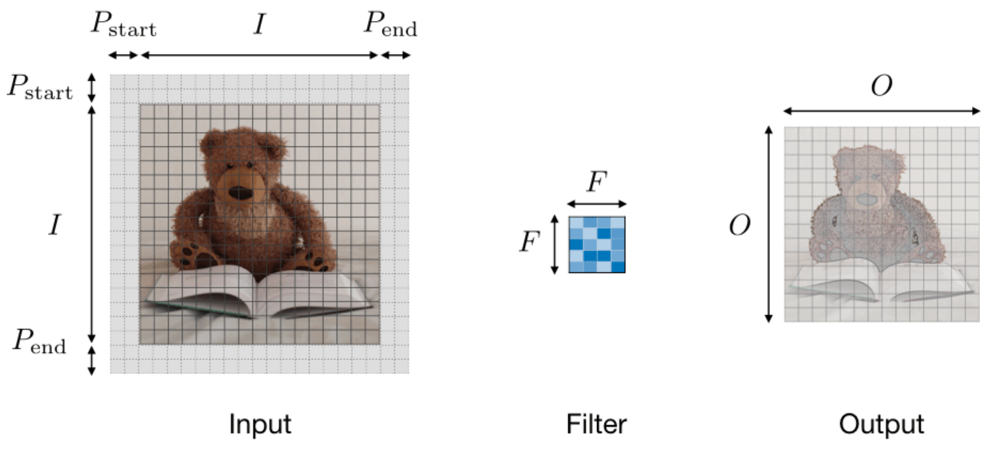


Figure 2.9 – Filter's parameters
Source: https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

— **Stride:** In convolution or pooling, the stride S denotes the number of pixels by which the window moves after each operation.

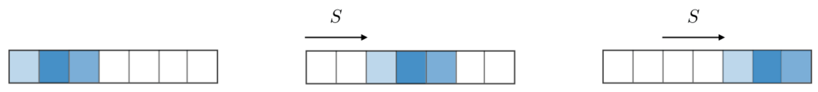


Figure 2.10 – Stride representation
Source: https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

— **Padding:** Zero-padding consists of adding P zeroes to each side of the boundaries of the input. There are mainly three mode in padding:

- **Valid:** In this mode there is no padding so we will drop the last convolution if dimensions do not match.
- **Same:** This mode of padding has a purpose of creating a feature map of size $\lceil \frac{I}{S} \rceil$ where I the input size and S is the stride.
- **Full:** In this mode we aim to have the maximum padding such that end convolutions are applied on the limits of the input.

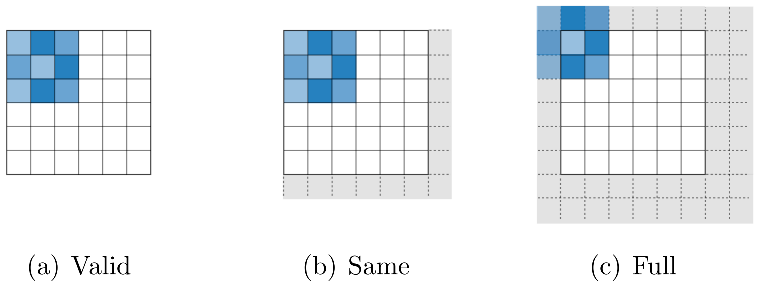


Figure 2.11 – Different padding modes
Source: https://stanford.edu/ shervine/teaching/cs-230/

2.3.3 Regularization for Deep Learning

2.3.3.1 Motivation

Deep neural network is powerful tool to predict a searched output. But, somtimes using large networks with lots of parameters could cause overfitting, where the model fit too closely the set of data poits, and give false predictions. Overfitting appears when the training error is small and the testing error is large. To deal with this problem, we should only use some of the techniques below.

2.3.3.2 Dropout

One of the most used method to reduce overfitting in a given model is dropout. The dropout operation is used at each training stage where individual nodes are either dropped out of the net with probability $1 - p$ or kept with probability p, so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed[13]. (see figure 2.12)

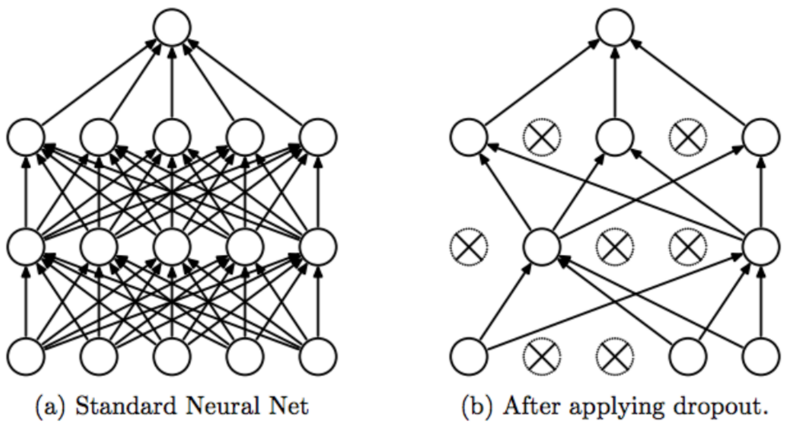


Figure 2.12 – Neural network architecture before and after dropout
Source: https://towardsdatascience.com/machine-learning-part-20-dropout-keras-layers-explained

The Dropout in a given neural network model could be described as below. Consider a neural network with L hidden layers. $z^{(l)}$ denote the input vector into layer l, $y^{(l)}$ denote the output vector into layer l. W^l and b^l are the weights and biases at layer l. The feed-forward operation of a standard neural network can be described as (for $l \in 0, \ldots, L-1$ and any hidden unit i)

$$z_i^{(l+1)} = W_i^{(l+1)} y^{(l)} + b_i^{(l+1)},$$

$$y_i^{(l+1)} = f(z_i^{(l+1)})$$

where f is the activation function. With dropout, the feed-forward operation becomes:

$$r_j^l \sim Bernoulli(p),$$

$$\tilde{y}^{(l)} = r_j^l * y^{(l)}$$

$$z_i^{(l+1)} = W_i^{(l+1)} \tilde{y}^{(l)} + b_i^{(l+1)},$$

$$y_i^{(l+1)} = f(z_i^{(l+1)}).$$

Where $*$ denotes an element-wise product and $r^{(l)}$ is a vector of independent Bernoulli random variables each of which has probability p of being 1[13]. For the backpropagation operation, the derivatives of the loss function are backpropagated through the sub-network. At test time, the weights are scaled as $W_{test}^{(l)} = pW^{(l)}$.

2.3.3.3 Batch Normalization

Batch normalization is one of the most recent innovations in optimizing deep learning. It is a technique used to improve the speed, the performance and the stability of the ANN. It is a step of hyperparameter γ, β that normalizes the batch x_i. The correction of the batch is done as follows:

$$x_i = \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

Where μ_B is the mean of the batch and σ_B is the varriance.

During training, this technique forces the network to periodically change its activations to zero mean and unit standard deviation, which works as a regularizer for the network, speeds up training, and makes it less dependent on parameter initialization [14].

2.3.3.4 Early stopping

Early stopping is used to precise the number of iterations that could be done before the network begins to overfit. The validation error normally decreases during the initial phase of training, as does the training set error. However, when overfitting begins happening, the validation error typically starts rising. Early stopping simply consists in stopping training when the validation error increases for a specified number of iterations (see figure 2.13).

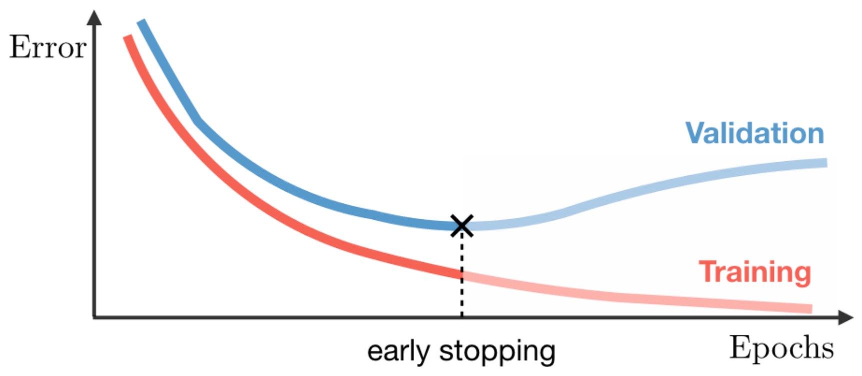


Figure 2.13 – Early stopping process
Source: https://stanford.edu/ shervine/teaching/cs-230/

2.3.3.5 Data augumentation

The best way to to make the model generalize better is to train it on more data. However, in practice the amount of data is limited. One proposed solution is to use data augmentation by creating fake data. This technique is usually done by taking images and make copies of them by scaling, cropping, rotating, ... these images.

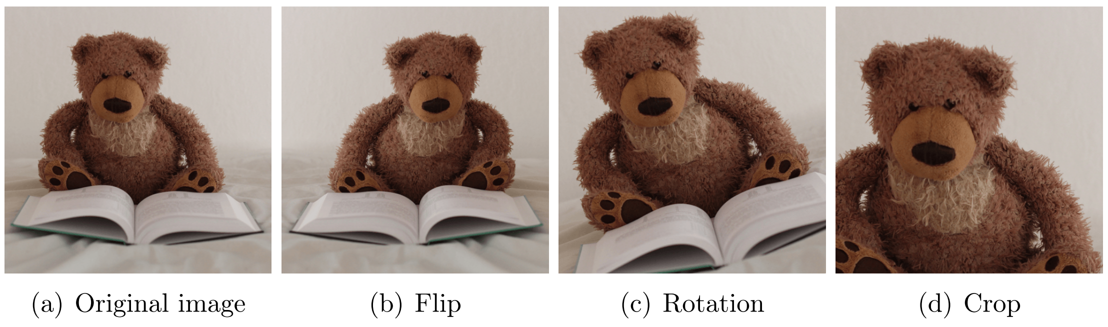


Figure 2.14 – Examples of data augumentation
Source: https://stanford.edu/ shervine/teaching/cs-230/

2.4 Blob detection

Blob detection is used in computer vision to detect regions in a digital image that differ in properties, such as brightness or color, compared to surrounding regions.

We will define the 3D version of two different algorithms for blob detection.

Let $f(x,y,z)$ the input image and $G(x,y,z,\sigma^2)$ a Gaussian kernel.

$$G(x,y,z,\sigma^2) = \frac{1}{(\sqrt{2\pi}\sigma)^3}\exp(-\frac{x^2+y^2+z^2}{2\sigma^2})$$

The convolution of the input image with the gaussian kernel is defined by:

$$L(x,y,z,\sigma^2) = G(x,y,z,\sigma^2) * f(x,y,z)$$

2.4.1 The Laplacien of Gaussian (LoG)

This approach consist of applying a scale-normalized Laplacian operator to create a multi-scale blob detector:

$$\nabla^2 L_{norm} = \sigma^3(L_{xx} + L_{yy} + L_{zz})$$

The selection of interest points $(\hat{x},\hat{y},\hat{z})$ and scales $\hat{\sigma^2}$ is performed using:

$$(\hat{x},\hat{y},\hat{z};\hat{\sigma^2}) = \text{argmaxminlocal}_{(x,y,z;\sigma^2)}((\nabla^2_{norm}L)(x,y,z;\sigma^2))$$

The radius of the detected blob will be $r = \sqrt{3}\sigma$ (expressed in voxel). This is the most accurate and slowest approach [15].

2.4.2 The Difference of Gaussian (DoG)

This method is inspired from the fact that $L(x,y,z,\sigma^2)$ satisfies the diffusion equation $\partial_{\sigma^2}L = \frac{1}{2}\nabla^2 L$.

So, the Gaussian operator $\nabla^2 L(x,y,z,\sigma^2)$ can be computed as the limit case of the difference between two Gaussian smoothed images.

$$\nabla^2_{norm}L(x,y;\sigma^2) \approx \frac{\sigma^2}{\Delta\sigma^2}\left(L(x,y;\sigma^2+\Delta\sigma^2) - L(x,y;\sigma^2)\right)$$

And we use the same operations to detect searched points and scales. Also, this is a faster approximation of LoG approach

Conclusion

In conclusion, this chapter proposed the necessery notions and terms nedded to deal with the problem of lung cancer detection. After defining the fundamentals about medical imaging, deep learning and image segmentation, we are goining to detail the proposed solution for lung cancer nodules generation.

Chapter 3

Proposed approach

Introduction

In this chapter, we will detail the proposed solution by explaining the different steps to detect nodules' candidates. Also, we will explain the proposed methods to reduce computation and optimize the model.

3.1 The general approach

The work done in this interndhip is a the first part of a project for detecting lung cancer from a given CT scan. The first step to do some preprocessing to minimize the calculation done to get the segmented scan and then extract the needed nodules. In addition to the preprocessing step, we propose to segment the lung from the CT scan to do the computation only on this specific zone; we don't need to apply nodules segmentation on bones or others tisues outside the lungs. As we see the the diagram 3.5, results found in the first step will be used to do nodules classification and false positive reduction in the next step.

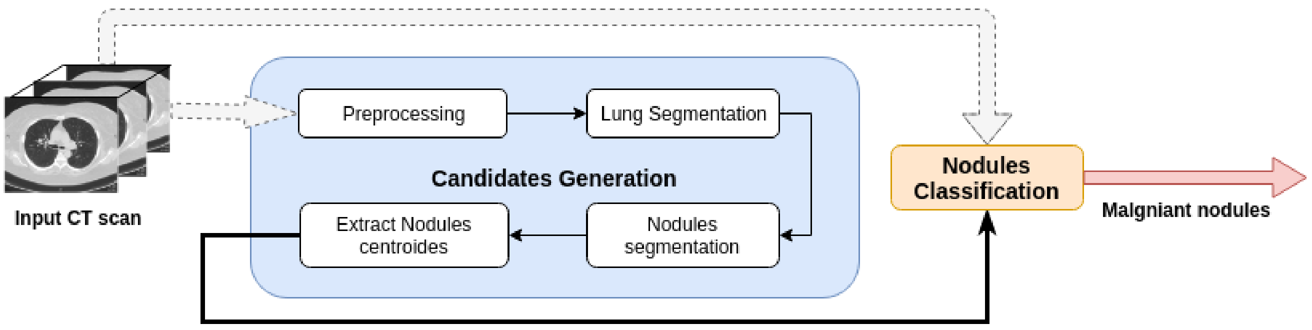


Figure 3.1 – Proposed Diagram for the given problem

3.2 Dataset

The dataset used in this project is LUNA16 [16]. It is data provided during the challenge LUNA16 in 2016 and it is a part of a public dataset called LIDC-IDRI database. In the LUNA16 data, we excluded slices thickness greater than 2.5 mm since pulmonary nodules could be very small so a thin slice should be chosen. Furthermore, scans with inconsistent slice spacing or missing slices were also excluded. This led to 888 CT scans, with a total of 36,378 annotations by radiologists. In this dataset, only the annotations categorized as nodules$\geq$ 3mm are considered relevant, as the other annotations (nodules $\leq$ 3mm and non-nodules) are considered irrelevant for lung cancer screening protocols[17]. Nodules found by different radiologists that were closer than the sum of their radius were merged. In this case, position and diameters of these merged annotations were averaged. This results in a set of 2290, 1602, 1186, and 777 nodules annotated by at least 1, 2, 3 or 4 radiologist(s), respectively. We will consider only nodules annotated at least by 3 radiologists. The LUNA16 data is mainly composed of two parts: the first one is a set of CT scans for different patients and the second is a list of nodules and irrelevant nodules for each patient.

3.2.1 CT scans

This part of the dataset is divided into 10 subsets. Each subset contains CT scans that were formatted '.mhd' and '.raw' . Where the ' .mhd' file contains some metadata about the scan extracted from the patient (see table 4.1). The '.raw' file is the binary format of the datapixel.

Field	Meaning	Example
ObjectType	Object type stored	Image
NDims	Dimension of the image	3
BinaryData	Precise if the data is binary or not	True
BinaryDataByteOrderMSB	Using MSB or not	False
CompressedData	Compressed data or not	False
TransformMatrix	Transformation matrix	[1 0 0 0 1 0 0 0 1]
Offset	The origin of the image	[-127.75 -277.75 -351.5]
CenterOfRotation	Center of rotation of the scan	[0 0 0]
AnatomicalOrientation	Orientation of image plane respect to the patient's axes of coordinates	RAI
ElementSpacing	Gap between voxels (spatial resolution)	[0.7 0.7 2.5]
DimSize	Size of the scan (number of slices in each direction)	[512 512 171]
ElementType	type of element values in MetaImage file.	MET_SHORT
ElementDataFile	Name of the of the binary data file	1.3.6.1.4.1.14 ... .raw

Table 3.1 – Information about the CT scan

3.2.2 Annotations

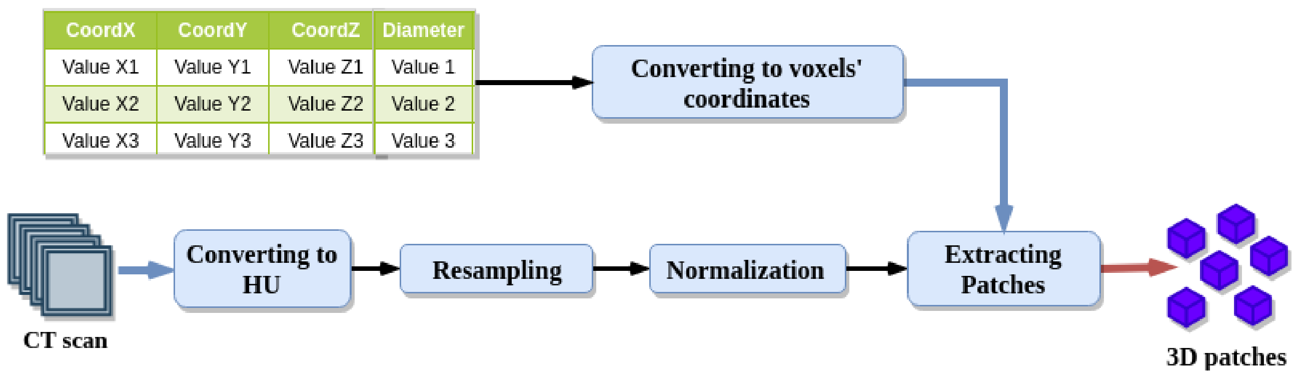
This part is formed of three files:

1. The annotation file is a csv file that contains one finding per line. Each line holds the SeriesInstanceUID of the scan, the x, y, and z position of each finding in world coordinates; and the corresponding diameter in mm. The annotation file contains 1,186 nodules. The follwing diagram
2. The second is also a csv file that contains irrelevant nodules (non-nodules, nodules with diameter less than 3 mm, and nodules annotated by only 1 or 2 radiologists). This file is formed of 35,192 lines. Each line holds the the SeriesInstanceUID of the scan, the x, y, and z position of each finding in world coordinates; and the corresponding diameter in mm if less than two radiologist annotated as nodules otherwise we put -1 as diameter.
3. Candidates file: this one provides list of nodules candidates to be used in the false positive task. These candidates are generated using some existing algortithm [18][19][20]. This file contains 551,065 candidates.

3.3 Preprocessing

3.3.1 Motivation

The preprocessing is mainly used to reduce calculation and assure that all our input data will have the same characteristics. There some operations to be done during this step (see figure 3.2).

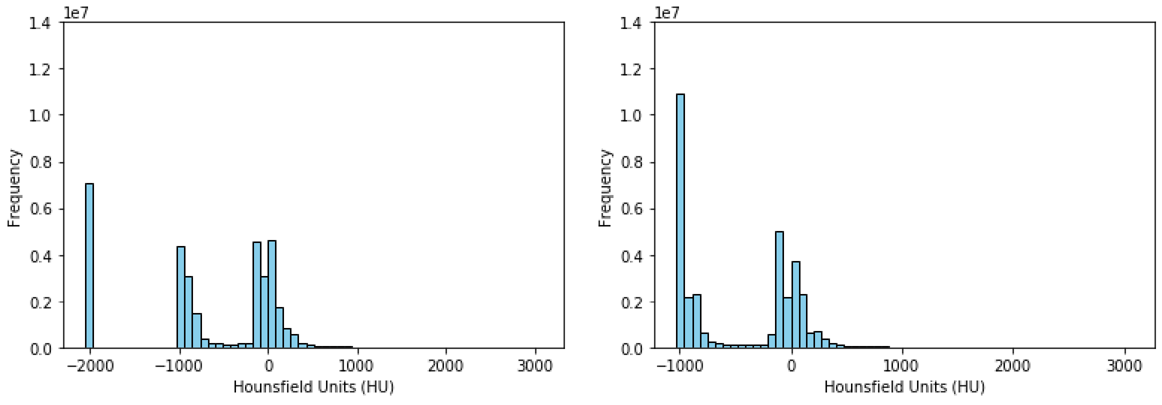


CoordX	CoordY	CoordZ	Diameter
Value X1	Value Y1	Value Z1	Value 1
Value X2	Value Y2	Value Z2	Value 2
Value X3	Value Y3	Value Z3	Value 3

Figure 3.2 – Preprocessing Digram to extract the training patches

3.3.2 Converting to HU

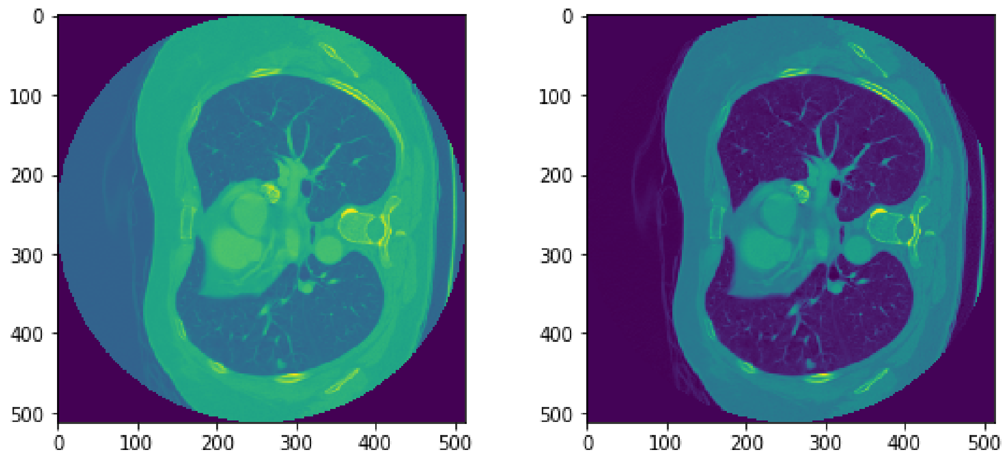
The first thing we should do is to check to see whether the Houndsfeld Units are properly scaled and represented. To do, we need to represent the histogram of some CT scans and make some interpretations.



(a) Initial histogram (b) Histogram after correction

Figure 3.3 – Histogram of all voxels in the CT scan

After displying the histogram of some CT scans and according to the table **??**, we could say that there is lots of air in each CT scan, There is some lung, there is some soft tissue, muscle, liver and some fat also. We could see that there is also some bit of bone. The most strange thing in the histogram is the bar at -2000. This value is explained by the fact: Some scanners have cylindrical scanning bounds, but the output image is square. The pixels that fall outside of these bounds get the fixed value -2000 [21]. The first step is setting these values to -1000, which corresponds to air.



(a) Original slice (b) Result after correction

Figure 3.4 – Resut of removing -2000 values

3.3.3 Resampling

Having different spacing for each scan is a problematic for automatic analysis. The common solution to this problem is resampling all the dataset to the same resolution.

Resampling the CT scan will transform it to an isotropic data to facilitate the visualization and the interpretation of the data. Each scan was rescaled so that each voxel represented an volume of 1x1x1 mm.

3.3.4 Normalization

The HU values in our CT scan are in a range from -1000 to 2000. The values that are interesting for us are values between -1000 and 400, anything above 400 are not interesting since these are bones with different radiodensity. Then, to normalize our data we apply this formula:

$$Scan = \frac{Scan - MinBound}{MaxBound - MinBound}$$

Where $MinBound = -1000$ and $MaxBound = 400$.
Then we apply a thresholding to get all values between 0 and 1.

3.3.5 Patches Extraction

In order to train our segmentation network, we need to extract patches with specific dimension. The dimension of these patches are chosen by taking into consideration the size of nodules which varies from 3 mm to 32 mm.

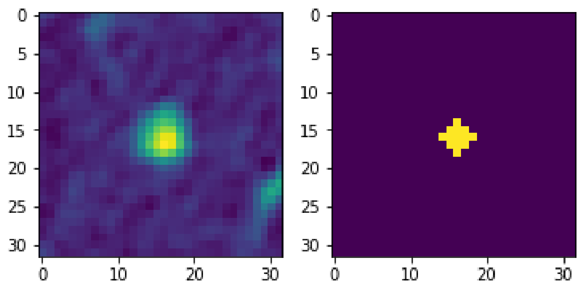


Figure 3.5 – Example of an extracted patch and his ground truth

In order to extract these patches, we need first to convert the given nodules real world's coordinates to voxeles' coordinates by applying this formula:

$$Voxel\ Coords = \frac{real\ world\ Coords - Origin}{Spacing}$$

After this transformation, we use the coordinates obtained to crop patches from images where the nodule will be in the center of the cube. For the ground-truth patches, we create the mask of all the CT scan then we crop the cubes using the obtained coordinates.

During the testing, we need to add some padding for some scans having the size multiple of the size of a patch.

3.4 Lung segmentation

This pre-processing step involves the segmentation of the lung volume from the surrounding tissues in the CT scan. The mask obtained from this segmentation is used to ensure that nodule detection is performed within the lung volume only.

This process has the two main advantage of reducing computation time and preventing the possible detection of false positive structures in regions of the image outside the lungs.

There are several techniques to do such work, we have chosen a simple one since the ultimate goal is detecting nodules. So, this problem was carried out using an algorithm that uses thersholding and some morphological operations applied on each slice in the CT scan.

1. Convert the slice to binary image: Using thresholding, we transform the input slice to a binary image. Values less than -400 will be set to ones and the rest will be set to zeros (see figure 3.6).

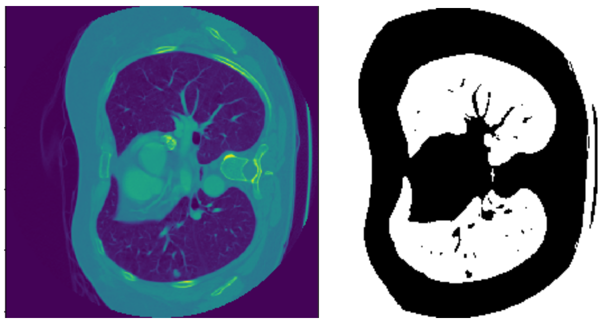

(a) Original slice (b) Binarization

Figure 3.6 – Step 1

2. Remove the region connected to the border of the image: as we can see in the first step, there are three big regions one of them is connected to the border of the image and should be removed to keep only the needed regions.

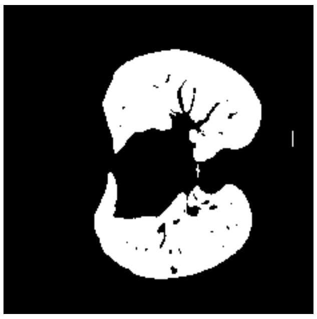
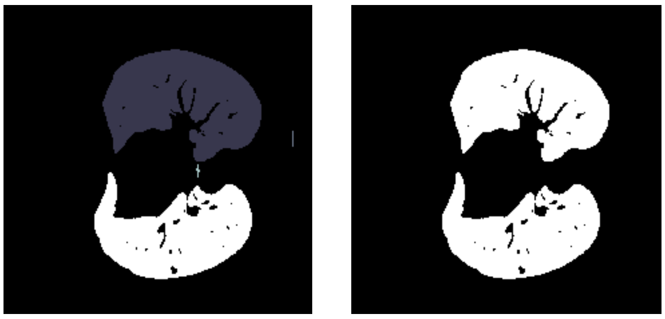


Figure 3.7 – Step 2

3. Label the image: In this step, first, we will detect connected regions and assign the same integer value to each one. Then, we will keep only the two largest regions.

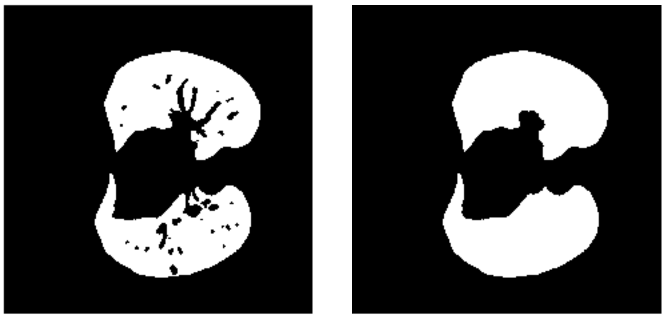


(a) Labeling regions (b) The two regions kept

Figure 3.8 – Step 3

4. Treatment of some nodules' cases:

   There are some nodules that will be attatched to blood vessels. To separate them we will use binary erosion (see appendix D) with a disk of radius 2. Some of the nodules are attached to the lung wall. To keep them in the two regions, we use a closure (see appendix D) with a disk of radius 10.



(a) Remove nodules attatched to blood vessels (b) Keep nodules attatched to lung wall

Figure 3.9 – Step 4

5. Fill in the small holes inside the binary mask of lungs:

   In some cases, there are small holes that still inside the mask of lungs, to remove them we detect first the edges of these holes using a Roberts edge operator, then, we fill them.

   The algorithm used for filling consists in invading the complementary of the shapes in input from the outer boundary of the image, using binary dilations. Holes are not connected to the boundary and are therefore not invaded [22].

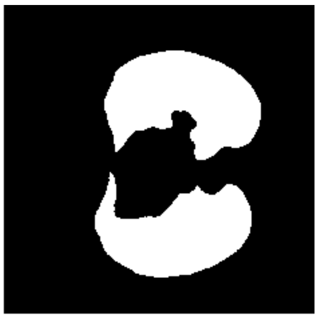


Figure 3.10 – Step 5

6. Superimpose the binary mask on the input image: The final step is to superimpose the created binary mask on the input image.

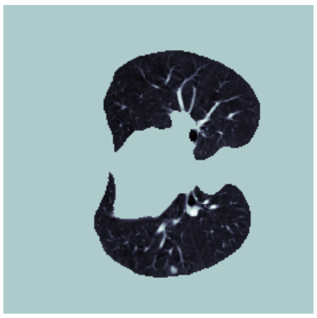


Figure 3.11 – Step 6

3.5 Nodules segmentation

In this section, we first explain the motivations behind using a modified version of UNet for nodules segmentation. Then, we will present the metrics used and the loss function used to train the proposed network.

3.5.1 Motivation

Based on the review [4] and the stat-of-art in medical imaging segmentation, UNet showed great results in segmenting nodules in lung cancer. Also, after seeing the results of the LUNA16 competition organized in 2016, we decided to modify the UNet architecture and employ 3D convolution to explore more this path.

3.5.2 Neural Network architecture

Inspired by the U-Net [23], we propose a network that exploits 3D up-sampling to restore input resolution from the pooled feature maps for nodule object segmentation. The proposed network takes patches of size $X \times Y \times Z$ as input to feed them into three down-sampling blocks. Each block consists of two consecutive convolution filters with size $3\times3\times3$, stride 1 and padding to reserve the input resolution, followed by the $2\times2\times2$ max pooling layer to reduce spatial resolution by 2. The coarsest resolution features are convoluted with $3 \times 3 \times 3$ filters twice to learn high-level nodule semantics. Then the resulting feature maps are feed into consecutive up-sampling blocks, with each block up-sampling features by 2 to finally restore the input resolution. For finer restoration of spatial resolution, we concatenate the up-sampled features with down-sampled features at the same spatial resolution. Finally, a $1 \times 1 \times 1$ convolution is used to reshape the feature maps to original input shape, before the sigmoid activation is applied to produce probability segmentation maps. We used ReLU for all others convolutions filters as activation function. The structure of the proposed architecture is shown in Figure 3.12.

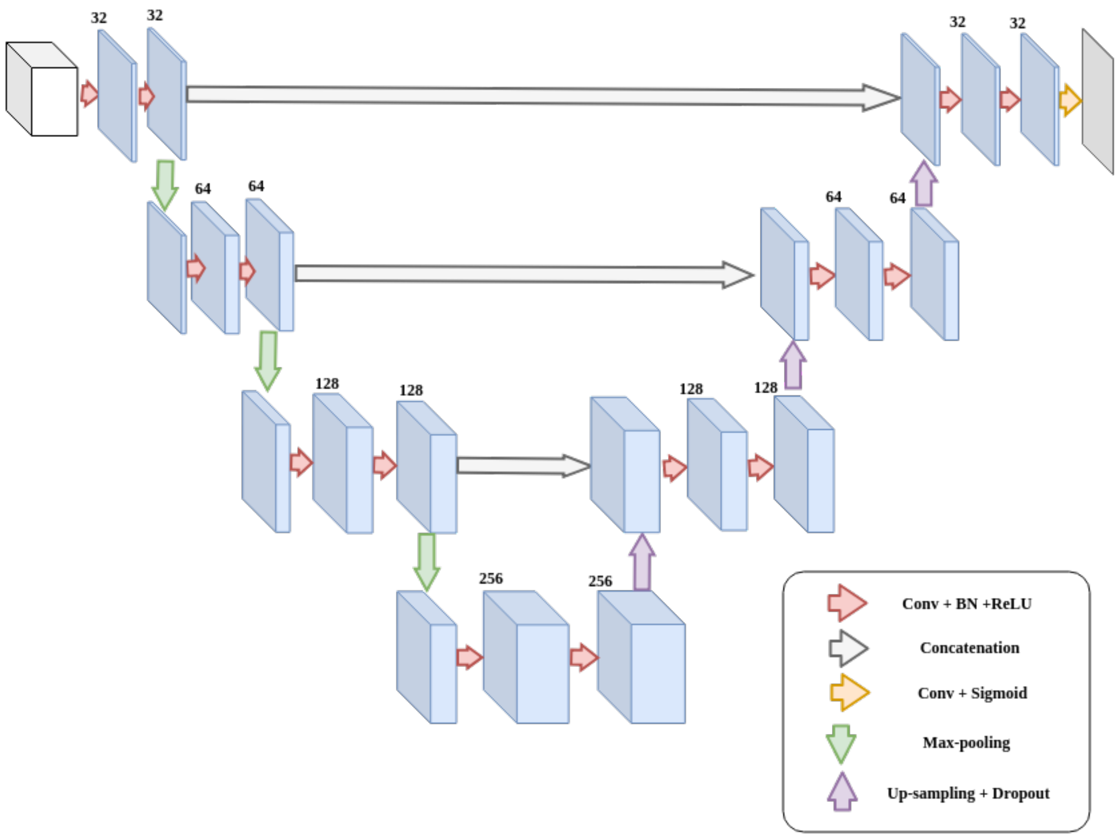


Figure 3.12 – Network architecture

3.5.3 Metrics and loss function

On of the most important terms to be fixed and well chosen are the evalution metrics and the loss function. To train the proposed network we have chosen the dice coefficient (DSC), the Intersection over Union (IoU) and as loss function we have chosen the dice loss.

- **Dice coefficient** [24]: It is also called F1-score. This metric is a interesting validation metric that measures the spatial overlap between two binary images. The values of this coefficient are in a range from 0 to 1.
  It is expressed as:
  $$DSC = \frac{2 \mid X \cap Y \mid}{\mid X \mid + \mid Y \mid} = \frac{2TP}{2TP + FP + FN}$$
  Where $\mid X \mid$ and $\mid Y \mid$ are the cardinalities of two sets X and Y. TP are the true positives, FP false positives and FN false negatives.
- **Intersection over union**: also know as Jaccard index. It is a method to quantify the overlap between the target mask and the prediction output.
  $$IoU = \frac{\mid X \cap Y \mid}{\mid X \mid \cup \mid Y \mid} = \frac{TP}{TP + FP + FN}$$
  Where $\mid X \mid$ and $\mid Y \mid$ are the cardinalities of two sets X and Y. TP are the true positives, FP false positives and FN false negatives.
- **Dice loss**: The dice loss is a Loss function proposed by Fausto Milletari et al. in V-net [25] and originated from the dice coefficient. Dice loss is actually the reverse of a dice coeficient. Searching for a good similarity between the output segmentation and the ground-truth is related to getting a higher dice coefficent and the model needs to find the minumum of a certain loss function. So, the commonly used function in this case will be $1 - DSC$ or $-DSC$.

3.6 Exctract Nodules centroides

In this part, we tried to reconstruct the given segmentation of the whole CT scan to apply a filter that could give us centroids and diameter of possible nodules. The proposed solution is to use one of the proposed algorithms for blob detection such us LoG and DoG.

So, to extract the centroids of the segmented nodule, we need to fix a range between σ_{min} and σ_{max} where the standard deviation of the Gaussian kernels vary. Blobs found are local maximas detected by successively increasing the standard deviatition. All the detected blobs were stacked together in a cube.

The σ_{min} and the σ_{max} were fixed using the formulation below:

$$\begin{cases} d = 2 \times \sqrt{3} \times p \times \sigma \\ 3mm \leq d \leq 32mm \end{cases} \Rightarrow \begin{cases} \sigma_{min} = \dfrac{3}{2 \times \sqrt{3} \times p} \\ \sigma_{max} = \dfrac{32}{2 \times \sqrt{3} \times p} \end{cases}$$

where d is the diameter of the nodule and p the size of the voxel expressed in mm (in our case $p = 1mm$).

We will evaluate this step by calculating the average number of candidates detected per scan and the time needed to make this extraction. We need to know also the sensitivity [1] of the model by calculating the rate of the true positive detected among all the detected centroid candidates. All the coordinates and the diameters of the nodules are saved in a '.csv' file to use them after in the classification process.

Conclusion

Using a modified version of 3D-UNet to tackle the problem of lung cancer nodules detection will be a very interesting approach with use of diffrent metrics like dice coeficient and intersection over union. The next step will be to see the results and evaluate them to make some interpretations.

1. $sensitivity = \frac{TP}{TP+FN}$ where TP is True Positive and FN is False Negative

Chapter 4

Experimental results

Introduction

In this chapter, we will detail the experimental process and discuss the found results. First, we will speak about the most important tools used in this project. Then, we will define the hyperparameters and the option used to prevent overfitting in our model. And finally, we will present our results and compare them with other methods.

4.1 Tools

In this section we will define the main tools used to realize the project and implement it. We implemented the propsed method in python. The most important libraries used in this project will be described briefly below.

Keras

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. Being able to go from idea to result with the least possible delay is key to doing good research [26].

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a library mainly used in machine learning and deep learning [27].

SimpleITK

SimpleITK is a simplified, open-source interface to the United States National Library of Medicine's Insight Segmentation and Registration Toolkit. The SimpleITK image analysis library was used with python as programming labguage [28].

4.2 Model training

4.2.1 Hardware charcteristics

One of the most important terms to train the model are the hardware's characteristics. All experments were conducted on a desktop computer with the following specifications:

- Eight cores Intel(R) Core(TM) i7-3770 CPU with a frequency @3.40GHz
- 12 GB RAM
- GPU of NVIDIA GeForce GTX 1060 with a memory of size 6 GB .
- Hard drive for data with a size 256 GB.

4.2.2 Training

To train the proposed network architecture, we have to choose the size of the patches. The dimension chosen is 32x32x32 since the the biggest geiven nodules have a diameter less than 32 mm. We will test also a size of 40x40x40 to see the difference in the perfomance of the model.

In order to evaluate the whole model, we have choden to leave 10% of the scans to test (88 CT scans). The rest of the scans will be used to train and validate our neural network proposed for segmentation. After applying the preprocessing step and extracting patches, we save these patches into '.npy' format to accelerate the training. We decided to use a data generator with a cache algoithm to speed up data genaration (The algorithm used is LRU see appendix C). The rest of 90% scans will be used to extract centred patches. 80% of the generated patches were used for training and the rest to test and validate the network architecture.

At the training stage, we search to minimize the dice loss over mini-batch of size 10 of training samples using Adam optimization with an initial learning rate of 0.001 and decay of $1e-5$. The network was trained for 100 epochs. After testing different initializers, we find that the best to initialize weights of a layer was He uniform initialization [29] which draws elements from a uniform distribution within $[-limit, limit]$ where $limit$ is $\sqrt{(6/N_{unit})}$ where N_{unit} is the number of input units in the weight tensor.

4.3 Regularization

4.3.1 Data augumentation

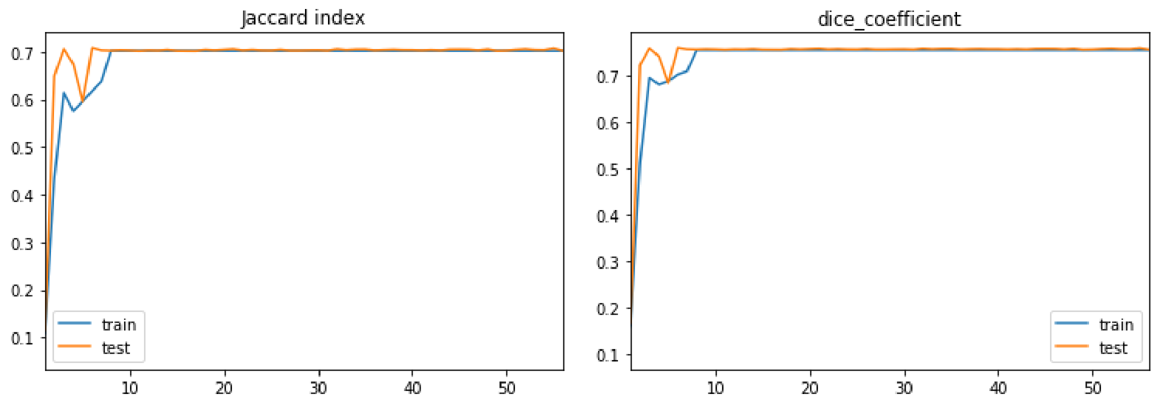
Given that only 1186 positive samples are availble to learn with, we need to apply some augumentation to get some fake data and increase the effective size of the training set. We do not need to apply these augumentationon the 35,192 irrelevant nodules since we are searching to create a sort of balance in the dataset. The created dataset We used four random rotations of $\pm 90°$ and $\pm 180°$, two random translations with random coefficients chosen from the set of integers $\{\pm5, \pm4, \pm3, \pm2, \pm1\}$, three flips around the three axes, two scalings with random scaling term chosen from the interval $[1.01, 1.25]$.

4.3.2 Dropout and Batch normalization

In the first test, and after using diffrent techniques such us batch normalization, decrease the size of some filters, we decided to add some dropout layers in our Network. We have chosen to add the dropout in the decoding part. The dropout was added in each block by a adding a dropout of 0.3 in the first block, 0.25 in the second one and 0.3 in the last block.

4.4 Results and interpretations

The training of the modified 3D-UNet was observed through the varriation of the loss function and the metrics (see figure 4.1).



(a) IoU curve (b) Dice coeffient curve

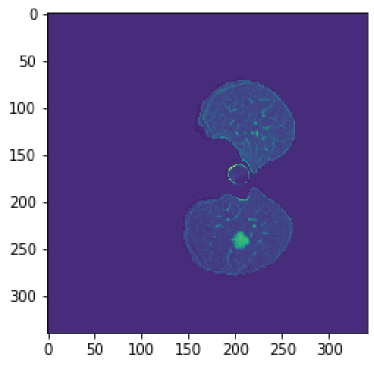
Figure 4.1 – Curves of training the network on 40x40x40 patches

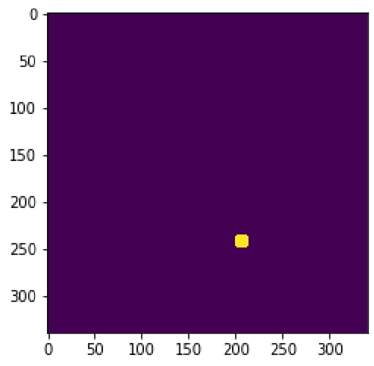
Table 4.1 presents the performance of the segmentation network:

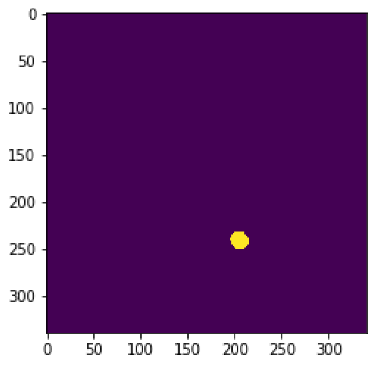
Size of samples	Dice coefficient	Jaccard index
32x32x32	78.91	71.81
40x40x40	76.13	70.26

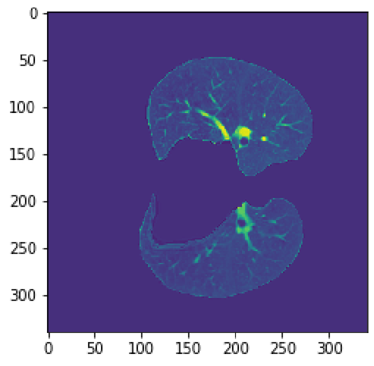
Table 4.1 – Results of the trained network

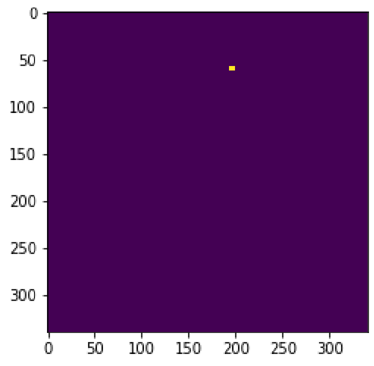
In the figure 4.2, we could see some results, the ground-truth images and the original slices.


(a) Original slice 1


(b) Ground truth 1


(c) Prediction 1


(d) Original slice 2


(e) Ground truth 2

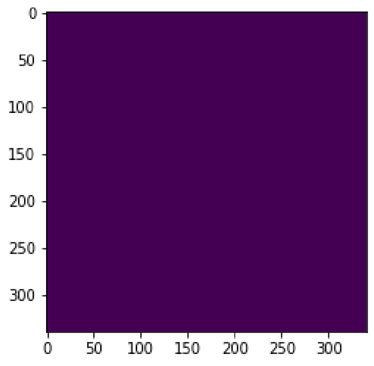

(f) Prediction 2

Figure 4.2 – Displaying results

As we can see in figure 4.2, there are some missed segmentations. These segmentation are essentially little nodules.

The centroids extraction was tested on the 88 CT-scans using LoG and DoG algorithms. The table 4.2 presents the performance of centroids extraction. The results show that the DoG is faster but detects less candidates than the LoG's method.

	Size of samples	**Num of candidates / scan**	**Time to generate candidates / scan**
DoG	32x32x32	52.6	36s
	40x40x40	3.9	35.63s
LoG	32x32x32	82	39s
	40x40x40	7.46	38.7s

Table 4.2 – Candidates centroids generation results

Table 4.3 summarizes the performance of our candidate detection algorithm in the two cases. The sensitivity was better in the case of 32x32x32.

	Sensitivity
32x32x32	92.1 %
40x40x40	87.62%

Table 4.3 – Sensitivity results

The senstivity of detection was caculated using the output of the trained network over 88 CT-scans left to test the whole model.

4.5 Results comparaison

We will compare the results found to some existing methods that used the same dataset. Since the best results aare found in the case of 32x32x32, we will compare them with two others approaches.

Based on an reticle published in the context of LUNA16 challenge [16], table 4.4 shows that the presented method give less candidates than the ETROCAD[30] method but with less sensitivity. However, our approach provide more candidates than M5L[31] method to classify them later on.

	Sensitivity %	Candidates / scan
ETROCAD[30]	**92.9**	333.0
M5L[31]	76.8	**22.2**
Our approach	92.1	42.3

Table 4.4 – Results comparaison

Conclusion

In this chapter, we have reported the results achieved by our model. We note that we managed to improve the state of the art results by applying a modified version of 3D-UNet to segment nodules and extract them using blob detection algorithms. The proposed model gave us acceptable results that could be enhenced in the future by using some post-proceccing operations.

Conclusion and Future perspectives

Throughout this graduation internship project, we have used convolutional neural network to detect lung cancer candidates from given CT scans. The main idea of the proposed system was to employ a modified version of 3D-UNet after making some pre-processing steps on the dataset. We achieved a sensitivity of 92.1% and we generated an acceptable number of candidates that could be used later on in the canssification part of the whole framework.

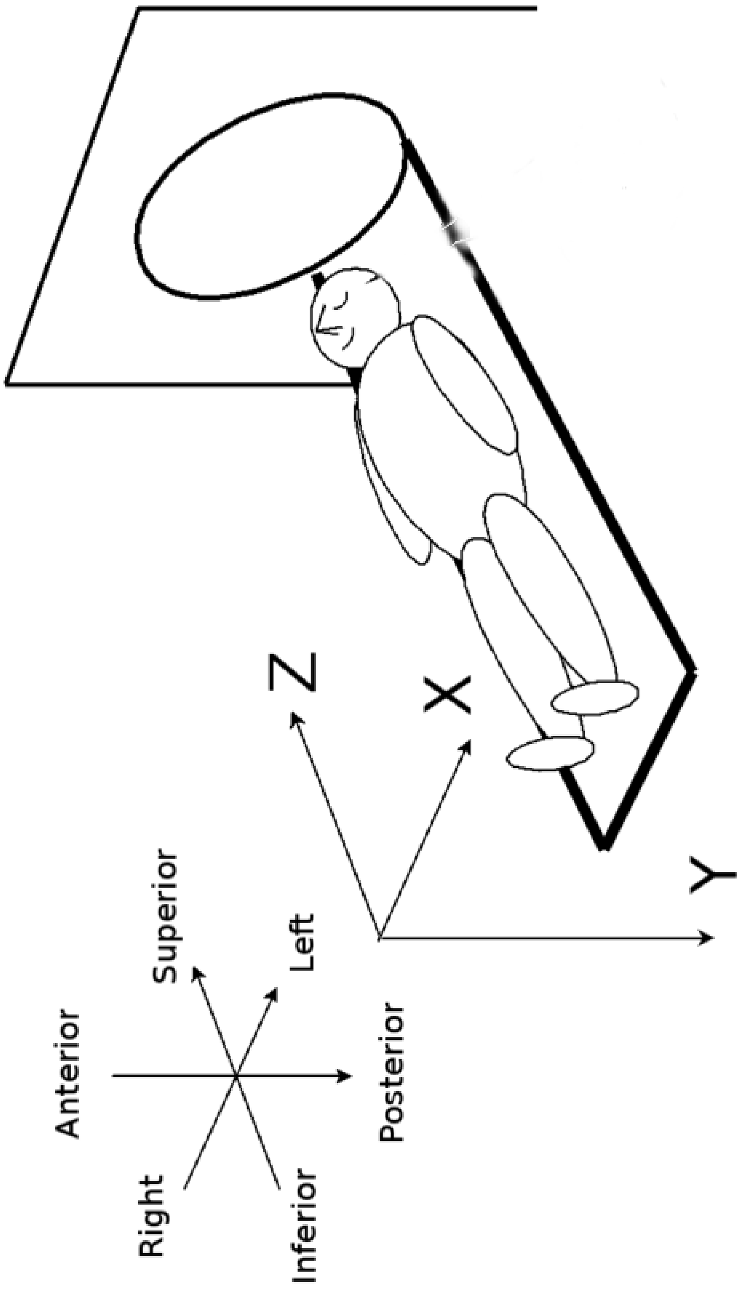
On a personal level, the period of this internship was extremely enriching since I had the opportunity to integrate a new team of researchers and to discover closely the research's environment. This internship also has allowed me to deepen my technical knowledge, especially in Python , Image processing and Keras programming while applying my academic and theoretical knowledge and learning new tips on deep learning. As far as the future prospects are concerned, there is a large room for improvement.

This project could be developed on the following axes :

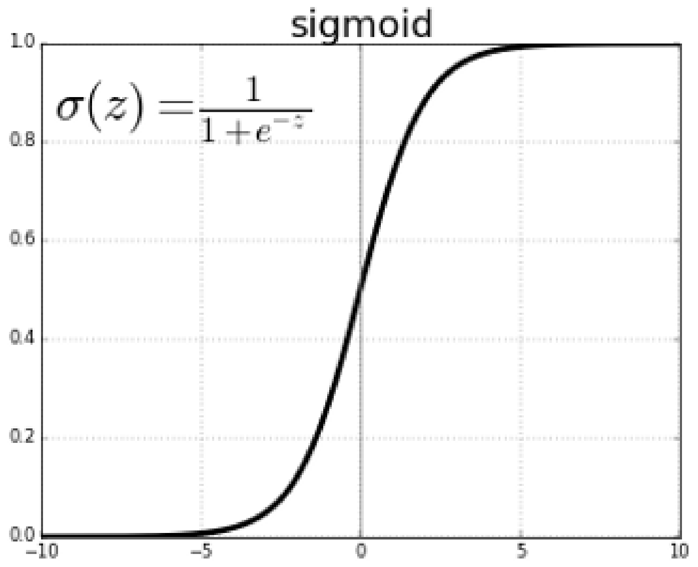
- Enhence the results by applying some post-processing methods.
- Test the proposed model on other datasets.
- Evalute the combination of the two parts of lung cancer detection.

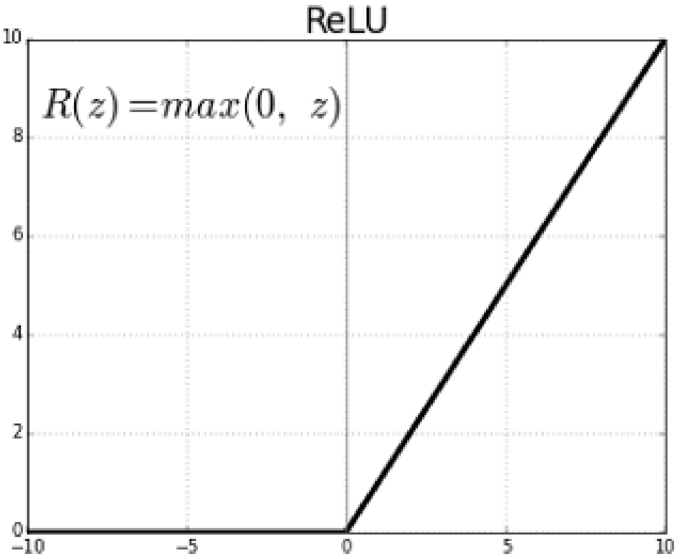
Appendix A

CT scanner Orientation



Appendix B
Sigmoid and ReLU functions





Appendix C
Least Recently Used (LRU)

The LRU is a cache algorithm that replaces the least recently used items first.
It requires keeping track of which items was used and when, and it is costly to make sure that the algorithm always discards the least recently used item. General implementation of this technique requires keeping "age bits" for cache-lines and track the "Least Recently Used" cache-line based on agebits.

Algorithm 2 LRU Algorithm

```
repeat

    if (current requested element is in cache) then
        Get its index
        Count to zero (indicate it is used very recently, higher the count of the most
least recently used element)
    else if (cache is full) then
        Get element with maximum count (LRU elemnt)
        Replace it with new element
        Reset count to zero
        Increment fault
    else
        Add new element to end of cache
        Increment the fault
        Increment top of cache
        Increment all the counts
    end if
    until (end of all requests) RETURN Number of faults
```

Appendix D
Morphological Operations

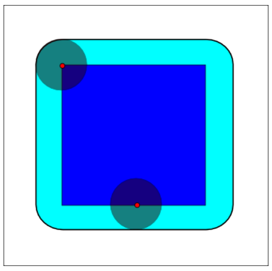
This part will give an over view about some morphological operations used in lung segmentation.We present here the basic and most used operations applied on binary images.
The binary image will be viewed as a subset of the integer grid Z^2. Let I (Dark blue square) be our image and S (Disk) is a structuring element centred on pixel ij by shifting S to S^{ij}.

Dilation

In dilation, position ij is included in the dilation of I by S if the intersection is non-zero:

$$I \oplus S = \{(i, j) : S^{ij} \vee I \neq 0\},$$

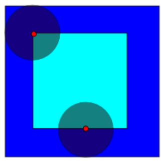
where $\vee$ is the logical AND operator.



Erosion

In erosion, position ij is included in the erosion of I by S whenever S is a subset of I:

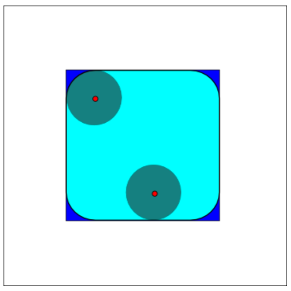
$$I \ominus S = \{(i, j) : S^{ij} \subseteq I\}$$



Opening

The opening of I by S is obtained by the erosion of I by S followed by dilation of the resulting image by S:

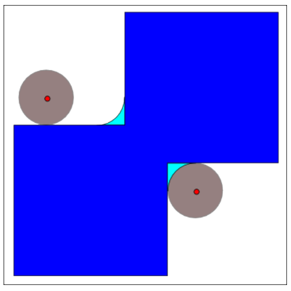
$$I \circ S = (I \ominus S) \oplus S$$



Closing

The opening of I by S is obtained by the dilation of I by S followed by erosion of the resulting image by S:

$$I \bullet S = (I \oplus S) \ominus S$$



Bibliography

[1] J. Doe and J. Doe, "A super interesting Article", *Journal of unreproducible Results*, 2015.

[2] "Reduced Lung-Cancer Mortality with Low-Dose Computed Tomographic Screening", *New England Journal of Medicine*, vol. 365, no. 5, pp. 395–409, 2011, PMID: 21714641. DOI: `10.1056/NEJMoa1102873`. [Online]. Available: `https://doi.org/10.1056/NEJMoa1102873`.

[3] U. de Moncton. (2019). University Website, [Online]. Available: `https://www.umoncton.ca/english/`.

[4] D. Riquelme and M. Akhloufi, "Deep Learning for Lung Cancer Tumors Detection".

[5] C. L. Epstein, *Introduction to the Mathematics of Medical Imaging, Second Edition*, second, ISBN: 089871642X, 9780898716429.

[6] L. G. Shapiro and G. C. Stockman, *Computer Vision.* 2001, ISBN: 0-13-030796-3. [Online]. Available: `http://nana.lecturer.pens.ac.id/index_files/referensi/computer_vision/Computer%20Vision.pdf`.

[7] D. Kaur and Y. Kaur, "Various image segmentation techniques: a review", *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 5, pp. 809–814, 2014.

[8] S. S. Haykin *et al.*, *Neural networks and learning machines/Simon Haykin.* New York: Prentice Hall, 2009.

[9] S. Ruder, "An overview of gradient descent optimization algorithms", *Journal of unreproducible Results*, 216.

[10] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016, `http://www.deeplearningbook.org`.

[12] W. Shi, J. Caballero, L. Theis, F. Huszar, A. Aitken, C. Ledig, and Z. Wang, *Is the deconvolution layer the same as a convolutional layer?*, 2016. arXiv: `1609.07009 [cs.CV]`.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting", *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *arXiv preprint arXiv:1502.03167*, 2015.

[15] T. Lindeberg, "Scale selection properties of generalized scale-space interest point detectors", *Journal of Mathematical Imaging and vision*, vol. 46, no. 2, pp. 177–210, 2013.

[16] A. A. A. Setio, A. Traverso, T. de Bel, M. S. Berens, C. van den Bogaard, P. Cerello, H. Chen, Q. Dou, M. E. Fantacci, B. Geurts, R. van der Gugten, P. A. Heng, B. Jansen, M. M. de Kaste, V. Kotov, J. Y.-H. Lin, J. T. Manders, A. Sóñora-Mengana, J. C. García-Naranjo, E. Papavasileiou, M. Prokop, M. Saletta, C. M. Schaefer-Prokop, E. T. Scholten, L. Scholten, M. M. Snoeren, E. L. Torres, J. Vandemeulebroucke, N. Walasek, G. C. Zuidhof, B. van Ginneken, and C. Jacobs, "Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The LUNA16 challenge", *Medical Image Analysis*, vol. 42, pp. 1–13, 2017. DOI: `10.1016/j.media.2017.06.015`. [Online]. Available: `https://doi.org/10.1016%2Fj.media.2017.06.015`.

[17] "Reduced Lung-Cancer Mortality with Low-Dose Computed Tomographic Screening", *New England Journal of Medicine*, vol. 365, no. 5, pp. 395–409, 2011. DOI: `10.1056/nejmoa1102873`. [Online]. Available: `https://doi.org/10.1056%2Fnejmoa1102873`.

[18] K. M. B. van Ginneken; A.M.R. Schilham; B.J. de Hoop; H.A. Gietema; M. Prokop, "A large-scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbour classification", *Medical Image Analysis*, vol. 13, pp. 757–770, 5 2009, ISSN: 1361-8415. DOI: `10.1016/j.media.2009.07.001`. [Online]. Available: `http://doi.org/10.1016/j.media.2009.07.001`.

[19] E. M.T.T.S.E.T.d.J.P.A.K.J.-M.O.M.d.K.H.J.P.M.S.-P.C.v.G. B. Jacobs Colin; van Rikxoort, "Automatic detection of subsolid pulmonary nodules in thoracic computed tomography images", *Medical Image Analysis*, vol. 18, pp. 374–384, 2 2014, ISSN: 1361-8415. DOI: `10.1016/j.media.2013.12.001`. [Online]. Available: `http://doi.org/10.1016/j.media.2013.12.001`.

[20] C. G.J.v.G. B. Setio Arnaud A. A.; Jacobs, "Automatic detection of large pulmonary solid nodules in thoracic CT images", *Medical Physics*, vol. 42, pp. 5642–5653, 10 2015, ISSN: 0094-2405. DOI: 10.1118/1.4929562. [Online]. Available: http://doi.org/10.1118/1.4929562.

[21] (). Medical Image Analysis with Deep Learning — I, [Online]. Available: https://medium.com/@taposhdr/MedicalImageAnalysiswithDeepLearningâĂŤI-TaposhDutta-Roy-Medium.

[22] E. Jones, T. Oliphant, P. Peterson, *et al.*, *SciPy: Open source scientific tools for Python*, [Online; accessed <today>], 2001–. [Online]. Available: http://www.scipy.org/.

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 234–241, 2015, ISSN: 1611-3349. DOI: 10.1007/978-3-319-24574-4_28. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-24574-4_28.

[24] A. A. Taha and A. Hanbury, "Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool", *BMC medical imaging*, vol. 15, no. 1, p. 29, 2015.

[25] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation", *2016 Fourth International Conference on 3D Vision (3DV)*, 2016. DOI: 10.1109/3dv.2016.79. [Online]. Available: http://dx.doi.org/10.1109/3DV.2016.79.

[26] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[27] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, Software available from tensorflow.org, 2015. [Online]. Available: http://tensorflow.org/.

[28] B. C. Lowekamp, D. T. Chen, L. Ibáñez, and D. Blezek, "The design of SimpleITK", *Frontiers in neuroinformatics*, vol. 7, p. 45, 2013.

[29] K. He, X. Zhang, S. Ren, and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”, *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015. DOI: `10.1109/iccv.2015.123`. [Online]. Available: `http://dx.doi.org/10.1109/ICCV.2015.123`.

[30] T. S.F.G.M.M.C. R. Riccardi Alessandro; Petkov, “Computer-aided detection of lung nodules via 3D fast radial transform, scale space representation, and Zernike MIP classification”, *Medical Physics*, vol. 38, p. 1962, 4 2011, ISSN: 0094-2405. DOI: `10.1118/1.3560427`. [Online]. Available: `http://doi.org/10.1118/1.3560427`.

[31] C. G.J.v.G. B. Setio Arnaud A. A.; Jacobs, “Automatic detection of large pulmonary solid nodules in thoracic CT images”, *Medical Physics*, vol. 42, pp. 5642–5653, 10 2015, ISSN: 0094-2405. DOI: `10.1118/1.4929562`. [Online]. Available: `http://doi.org/10.1118/1.4929562`.