CLASSIFICATION OF LUNG NODULES IN CT IMAGES USING CONVOLUTIONAL NEURAL NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GÖRKEM POLAT

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2018

Approval of the thesis:

CLASSIFICATION OF LUNG NODULES IN CT IMAGES USING CONVOLUTIONAL NEURAL NETWORKS

Submitted by **GÖRKEM POLAT** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tolga Çiloğlu
Head of Department, **Electrical and Electronics Engineering** _____

Assoc. Prof. Dr. Yeşim Serinağaoğlu Doğrusöz
Supervisor, **Electrical and Electronics Eng. Dept., METU** _____

Prof. Dr. Uğur Halıcı
Co-Supervisor, **Electrical and Electronics Eng. Dept., METU** _____

Examining Committee Members:

Prof. Dr. Nevzat Güneri Gençer
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Yeşim Serinağaoğlu Doğrusöz
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Uğur Halıcı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Murat Eyüboğlu
Electrical and Electronics Engineering Dept., METU _____

Assist. Prof. Dr. Görkem Saygılı
Biomedical Engineering Dept., Ankara University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Görkem Polat

Signature :

ABSTRACT

CLASSIFICATION OF LUNG NODULES IN CT IMAGES USING CONVOLUTIONAL NEURAL NETWORKS

Polat, Görkem

M. Sc., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Yeşim Serinağaoğlu Doğrusöz

Co-supervisor: Prof. Dr. Uğur Halıcı

January 2018, 101 pages

Recent studies have shown that lung cancer screening using annual low-dose computed tomography (CT) reduces lung cancer mortality by 20% compared to traditional chest radiography. Therefore, CT lung screening has started to be used widely all across the world. However, analyzing these images is a serious burden for radiologists. The number of slices in a CT scan can be up to 600. Therefore, computer-aided-detection (CAD) systems are very important for faster and more accurate assessment of the data. In this thesis, we proposed a framework that analyzes CT lung screenings using convolutional neural networks (CNNs) to reduce false positives. Our framework shows that even non-complex architectures are very powerful to classify 3D nodule data when compared to traditional methods. We trained our model with different volume sizes and showed that volume size plays a critical role in the performance of the system. We also used different fusions in order to show their power and effect on the overall accuracy. 3D CNNs were preferred over 2D CNNs because data was in 3D and 2D convolutional operations may result in information loss. The proposed framework has been tested on the dataset provided by the LUNA16 Challenge and got a sensitivity of 0.831 at 1 false positive per scan.

Keywords: Lung Nodule Detection, Computed Tomography, Convolutional Neural Networks, Deep Learning.

ÖZ

BT GÖRÜNTÜLERİNDE AKCİĞER NODÜLLERİNİN EVRİŞİMSEL SİNİR AĞLARI KULLANILARAK SINIFLANDIRILMASI

Polat, Görkem

Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Yeşim Serinağaoğlu Doğrusöz

Ortak Tez Yöneticisi: Prof. Dr. Uğur Halıcı

Ocak 2018, 101 sayfa

Son yapılan çalışmalar her yıl yapılan düşük dozlu bilgisayarlı tomografi (BT) taramalarının, geleneksel göğüs radyolojisine göre akciğer kanserinin erken tespitinde %20 daha iyi sonuç verdiğini göstermiştir. Bu sebeple akciğerin BT ile incelenmesi tüm dünyada yaygınlaşmaktadır. Fakat bu görüntülerin analiz edilmesi radyologlar için ciddi bir yüktür. Bir BT taramasındaki görüntü sayısı 600'e kadar çıkabilmektedir. Bu sebeple bilgisayar destekli tespit sistemleri görüntülerin daha hızlı ve daha doğru tanınması için çok önemlidir. Bu çalışmada evrişimsel sinir ağları (ESA) kullanılarak akciğer BT görüntülerini analiz eden ve yanlış-pozitifleri azaltan bir yöntem geliştirilmiştir. Sinir ağı modeli, farklı boyutlardaki girdiler ile denenmiş ve girdi boyutunun sistemin performansına olan etkisi gösterilmiştir. Ayrıca, bir çok modelden elde edilen sonuçlar değişik kombinasyonlarda bir araya getirilerek başarım artırılmış ve bu yöntemin gücü gösterilmiştir. Sınıflandırılacak bilginin 3 boyutlu olması ve veriyi 2 boyutlu işlemenin bilgi kaybına yol açmasından dolayı 3 boyutlu evrişimsel sinir ağları kullanılmıştır. Önerilen yöntem LUNA16 Yarışması tarafından sağlanan

veri seti üzerinde denenmiş ve tarama başına 1 yanlış pozitif oranında 0.831 duyarlılığına ulaşılmıştır.

Anahtar Kelimeler: Akciğer Nodül Tespiti, Bilgisayarlı Tomografi, Evrişimsel Sinir Ağları, Derin Öğrenme.

To my precious wife Ece for her patience and support,

Gülcan and Ali for being such good parents,

and my brother Uğur for his cheer.

ACKNOWLEDGEMENTS

First of all, I would like to thank to Assoc. Prof. Dr. Yeşim Serinağaoğlu Doğrusöz for her guidance, patience, and support in this thesis.

I would like to thank Prof. Dr. Uğur Halıcı for her assistance and valuable suggestions through this thesis. Without her, this thesis would take more time to complete.

I would like to express my thanks and appreciation to LUNA16 organizing team for their efforts to provide the data and answer questions.

I would like to thank to METU EEE Computer Vision Laboratory for providing me a workstation in order to run algorithms. Without this support, implementation of CNN architectures would be impossible to train.

I would like to express my sincere gratitude to my wife Ece for her patience. Although being a very social woman, she waived the social life around us and always supported me during this thesis.

My mother and father, Gülcan and Ali, were always supportive in every stage of my life. I present this thesis to them.

Finally, thanks to my cool brother Uğur for adding joy to my life.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF TABLES

TABLES

LIST OF ABBREVIATIONS

AdaGrad	Adaptive Gradient Algorithm
ADAM	Adaptive Momentum Estimation
ANN	Artificial Neural Networks
CAD	Computer Aided Detection
CNN	Convolutional Neural Networks
ConvNets	Convolutional Neural Networks
CT	Computed Tomography
FP	False Positive
FROC	Free-Response Receiver Operating Characteristic
k-NN	k-Nearest Neighbor
LDA	Linear Discriminant Analysis
ML	Machine Learning
MLP	Multilayer Perceptron
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
NLST	National Lung Screening Trial
NSCLC	Non-Small Cell Lung Cancer
PET	Positron Emission Tomography
RG	Region Growing
SVM	Support Vector Machines
TP	True Positive

CHAPTER 1

INTRODUCTION

1.1 Motivation and Overview

Lung cancer is a malignant lung tumor characterized by uncontrolled cell growth in the lung tissues. Lung cancer occurred in 1.8 million people and resulted in 1.6 million deaths worldwide in 2012 [1], which makes it the most common cause of cancer-related death in men and second most common in women after breast cancer [1]. The National Lung Screening Trial (NLST), a randomized control trial in the U.S. including more than 50,000 high-risk subjects, showed that lung cancer screening using annual low-dose computed tomography (CT) reduces lung cancer mortality by 20% compared to chest radiography [2]. Therefore, low-dose CT scanning programs have started to be implemented in the United States and other countries.

One of the major challenges of CT is that many images must be analyzed by the radiologists. The number of slices in a CT scan can be up to 600. Analyzing these enormous data is a serious burden for radiologists. Therefore, computer aided detection (CAD) systems are very important for faster and more accurate assessment of the data.

A CAD system for nodule detection generally consists of two steps: 1) nodule candidate detection, 2) false positive reduction. Candidate detection step aims to generate candidate points that are suspected of being nodule. High sensitivity is very important in this step; therefore, many false positives are also generated. High number of false positives is not a desired situation because it increases the number of candidate nodules to be observed by the radiologists. Therefore, false positive reduction stage

reduces the number of false positives among the candidates by keeping the same sensitivity.

Lung nodule detection using CAD systems has been a very active research area for the past two decades. Both research groups and companies have been developing algorithms in order to detect nodules accurately. One of the first fully automated computerized methods for the detection of lung nodules was proposed by Armato et al. [3] in 2001. They used morphological and gray level features in order to detect the nodules. They applied their algorithm to 43 cases and got a detection sensitivity of 70% with an average of 1.5 false-positives per case. There were not enough lung CT data in those years. Therefore, for the next couple of years, several studies used morphological image processing and unsupervised learning techniques [4] [5] [6] [7] [8] [9]. During the early days of the development of lung nodule detection systems, every researcher was using different datasets. In those datasets, CT image qualities, distance between slices, and morphological structures of the nodules were very different from each other. Therefore, it was hard to compare these algorithms. As a result, ANODE09 [10] and LUNA16 Challenges [11] were organized. In these challenges, participants worked on the same data. A comprehensive literature survey on algorithms and lung nodule detection systems has been presented in the Background Information chapter.

With the increasing processing power and new artificial intelligence (AI) algorithms, detection scores have been rising year by year. In the recent years, machine learning techniques showed significant improvements in detection accuracies but datasets that were used to train the algorithms are still insufficient. As the quality of CT images improves and more data are collected and labeled, these algorithms promise to give good results.

Recently, convolutional neural networks (CNNs) have become very famous in machine learning field due to their high performance. CNNs are made up of neurons that have learnable weights and biases. This algorithm is based on artificial neural network (ANN) structure, which is inspired by the biological neuron. One advantage of CNNs over traditional neural networks is that filters are learned by the system itself. CNN layer parameters consist of a set of learnable filters, which makes the system

adaptive for problems. By using convolution operation, these filters extract the spatial information in the input data. Therefore, CNNs have very good results in object detection, video analysis, voice recognition, natural language processing and medical image analysis [12] [13] [14]. Yet, CNNs usually require a large amount of training data in order to avoid overfitting.

Deep learning techniques were also applied on the classification of lung nodules. With the LUNA16 Challenge [15], which will be explained in detail later, teams applied deep learning techniques to candidate generation and false positive reduction steps. Since CNNs may have different architectures, there were many different models which were used in this challenge. Best model achieved the sensitivity of 0.848 at 1 false positive per scan. In this challenge, nodule candidates were ranging from 3 mm to 34 mm. Therefore, training patch sizes were different on all models. While some of the teams have used larger patches to cover all nodules, others have used smaller patches in order to get rid of noise. The team that got the best result used three different patch sizes and different model architectures for each of them. At the end of the challenge, it was observed that patch size was playing an important role in the detection accuracy of the algorithm. Yet, there was no objective study that compares the effect of different patch sizes to the result.

1.2 Objective of the Thesis

The aim of this thesis is building a CAD system with high sensitivity and low false positive rate by using CNN architecture. One of the most important criteria in this aim is the accuracy of the classification. A fast algorithm with low accuracy is not applicable to the clinical use because missing the true positives would be very harmful for the patients.

Another objective is to compare the effect of the input patch size on the classification performance. In the classification process, nodules are classified individually. In other words, whole CT data is not processed alone. Instead, nodules are extracted from the CT data and evaluated one by one. Yet, for the 3D CNN model, a fixed patch size

should be determined. This is a very crucial factor in the performance of model training. If different patch sizes focus on different characteristics, results of different models can be merged in order to generate better results. Therefore, ensemble of classifiers must be used at the end of the work in order to observe whether there is an improvement in the performance or not.

1.3 Scope

The dataset used in this work is provided by LUNA16: Lung Nodule Analysis Challenge [15]. The dataset consists of CT images. Each CT image has a varying number of slices. In the dataset, there is also a text file that consists of locations of the candidates and labels for indicating candidate class (nodule or non-nodule). Aim was to reduce the number of false positives in this dataset and compare the effect of input patch size to the performance. In order to achieve these goals, we created a 3D CNN model architecture. While forming the 3D CNN model, we also presented how different patch sizes affect the results and how ensemble of classifiers changes the overall accuracy. We compared algorithms that use different patch sizes for the sake of further studies.

1.4 Contributions

In this thesis, we have used 3D CNNs along with a preprocessing step and decision fusion at the end in order to classify pulmonary nodule candidates and reduce the number of false positives that were generated by the candidate detection algorithms. We got a sensitivity of 0.831 at 1 false positive per scan and 0.913 at 8 false positives per scan. The average sensitivity at 7 predefined false positive rates, which are 0.125, 0.25, 0.5, 1, 2, 4, and 8, was 0.786.

We have applied 3D CNNs successfully to lung nodule detection. By applying some preprocessing techniques and ensemble of classifiers at the end, using 3D CNNs outperforms the traditional image processing and machine learning tasks such as

kernels and morphological operations [10]. We have created a novel 3D CNN architecture that takes the candidate patches as an input and applies training over them. Our model has a lower complexity compared to other models so that its training and detection time is lower compared to other models [16]. We have applied data augmentation and mini-batch classification techniques in order to handle imbalances in the output classes. Our proposed framework is very generic and can be easily applied to the other 3D classification tasks such as breast, brain or liver nodules.

There are many 3D medical images that are generated from the magnetic resonance imaging (MRI) and CT. When classifying nodules, selection of the patch size plays an important role. In this thesis, we have showed how different receptive fields affect the detection result. In order to objectively compare the receptive fields, they were trained on similar network architecture. We have observed that, receptive field changes the result noticeably. This comparison shows that when there is a 3D classification task, using different receptive fields should be taken into account because different receptive fields may generate different result sets and they can be used to increase sensitivity by using ensemble of classifiers.

1.5 Organization of the Thesis

This thesis contains five chapters, which are the introduction, background information, proposed approach, experimental results, and conclusion.

In chapter 1, introductory information is presented to the reader. The motivation behind this work, an overview of the thesis, information about the organization of the thesis and contributions are presented in this chapter.

In chapter 2, medical information about lung cancer and background information about the detection methods have been explained in detail. On the medical side, the current state of lung cancer, statistics around the world, and related information about it have been presented. Why automated detection algorithms are needed is explained in this chapter. On the technical side, methods in the literature for detection of lung cancer and technical background on proposed framework have been explained. A

comprehensive literature survey on lung nodule detection is presented in this chapter. Powerful features of the proposed method compared to other algorithms have been described.

In chapter 3, proposed solution has been explained. Preprocessing steps, network structure, the value of parameters and hyper-parameters, and applied algorithm have been explained in detail.

In chapter 4, results of the experiments have been proposed. Performance metrics, Free-response Receiver Operating Characteristics (FROC) curves, and score tables have been presented. In this chapter, results of combination of different models are also presented.

In chapter 5, overall work is summarized. Insights gained during this work and future experiments that can be done on this topic have been discussed.

CHAPTER 2

BACKGROUND INFORMATION

In this chapter, background information on lung cancer and technical background on its detection have been explained. Anatomy of the lung, the current state of lung cancer around the world, its statistics and causes have been described. Literature survey on lung nodule detection systems and algorithms used so far have been explained. Later in the chapter, fundamentals of machine learning methods, the theory behind the neural networks, and convolutional neural networks have been explained.

2.1 Anatomy of the Lung

Lungs are a pair of large and spongy organs that are found in the thorax lateral to the heart and on the upper part of the diaphragm (Figure 1). Each lung is surrounded by a membrane that provides the lung with space to expand. The left and right lungs are slightly different in size and shape due to the heart which is located near the left lung. Therefore, left lung is slightly smaller than the right lung and consists of 2 lobes while the right lung has 3 lobes. Interior of the lungs is made of around 30 million sacks which are called the alveoli. Alveoli are lined with thin simple squamous epithelium that allows air entering the alveoli to exchange its gases with the blood passing through the capillaries.

The air, which contains oxygen and other gases, comes into the body through the lungs. In the lungs, the oxygen is moved into the blood-stream and carried through the body. Red blood cells collect the carbon dioxide and transport it back to the lungs, where it leaves the body when we exhale.

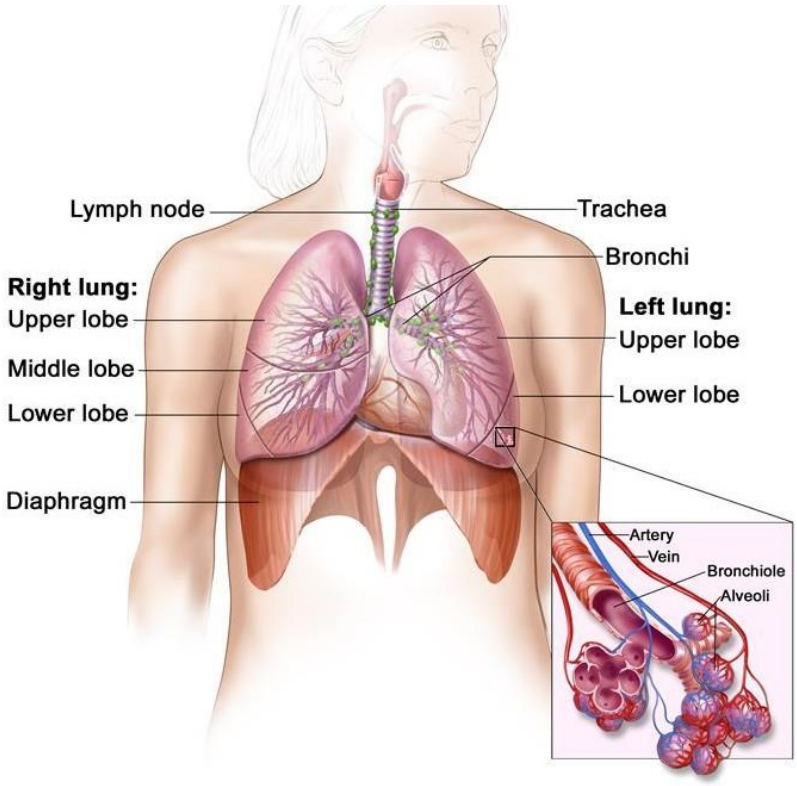

Figure 1: Anatomy of the respiratory system [17].

2.2 Lung Cancer

Lung cancer, like all cancers, is uncontrolled abnormal cell growth in tissues of the lung [18]. During the growth, cells are spread through the lung by the process of metastasis into nearby tissue or other parts of the body. This cell growth eventually forms a mass known as a tumor. There are three main types of lung cancer, which are non-small cell lung cancer (NSCLC), small cell lung cancer and lung carcinoid tumor. NSCLC is the most common type lung cancer, which constitutes about 85% of all lung cancers [19].

Lung cancer is the most common cause of cancer-related deaths worldwide. It is estimated that nearly 1.6 million people (19.4% of all cancer deaths) died because of lung cancer in 2012 [20]. Overall, the chance that a man will develop lung cancer in his lifetime is about 1 in 14; for a woman, it is about 1 in 17. Statistics on survival rates

in people with lung cancer depend on the stage of the cancer. According to the National Cancer Institute's SEER database [21]:

- The 5-year survival rate for people with stage 1A NSCLC is about 49%.
- The 5-year survival rate for people with stage 2A NSCLC is about 30%
- The 5-year survival rate for people with stage 3A NSCLC is about 14%
- The 5-year survival rate for people with stage 4 NSCLC is about 1%

The main cause of the lung cancer (85%) is long term-usage of tobacco [22]. Evidence shows that there is a strong correlation between usage of tobacco and lung cancer [22]. As seen in Figure 2, increase and decrease trends of lung cancer are highly parallel to the usage of the tobacco. Even the effect of the Great Depression [23] can be seen in the figure, when the consumption is decreased, lung cancer incidence is also decreased in the following years. Besides usage of tobacco, passive smoking, radon gas, genetic factors, and air pollution also cause lung cancer.

The most common symptoms of the lung cancer are:

- Cough (with blood),
- Shortness of breath,
- Chest pain,
- Wheezing,
- Difficulty in swallowing,
- Feeling tired or weak.

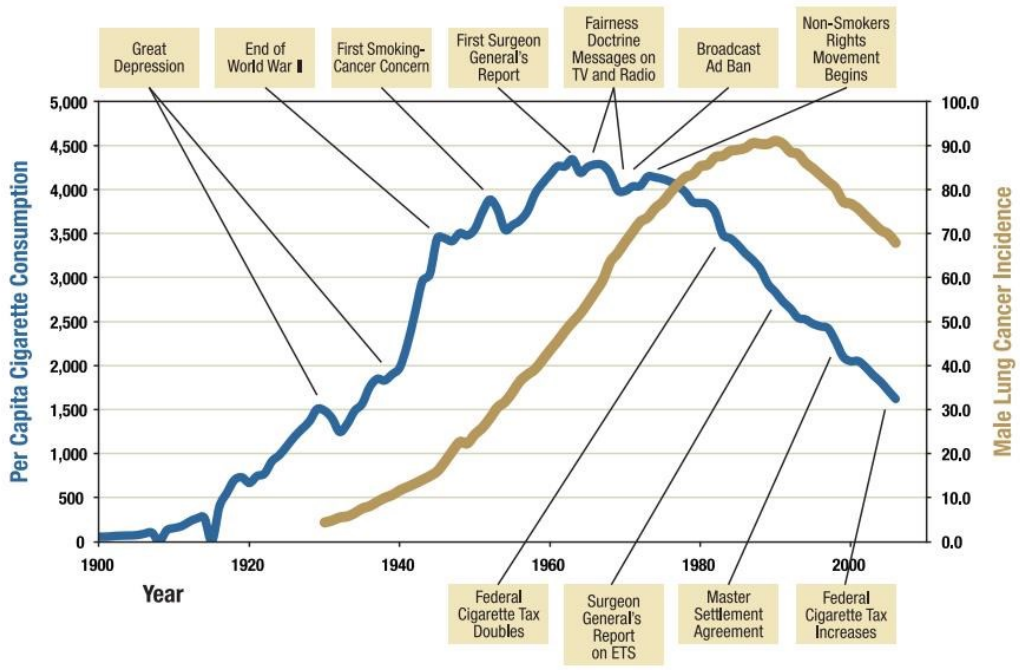

Figure 2: Tobacco usage vs. lung cancer incidence [22].

Usually, symptoms of lung cancer do not appear until the disease has spread to an advanced stage. There are very rare cases where there is an early diagnosis of lung cancer. Therefore, a robust screening and diagnosis systems are very important for early diagnosis. With imaging tests, doctors can look for suspected regions, examine possible signs of cancer or determine whether the treatment is working or not.

2.2.1 Screening Methods

When doctors want to assess the lung health for the first time, they generally order chest X-rays (Radiology). Chest X-rays expose the patient to a small amount of radiation and may reveal suspicious areas in the lungs, but this method is not capable of determining if these areas are cancerous. It is generally a 2D view from the back to front. Computed tomography (CT) uses X-rays to make detailed cross-sectional images of the body. Instead of one image, CT scanner takes multiple images (projections) and they are merged by the computer software. CT images are more detailed compared to a single X-ray image. In CT images, size, shape, and position of

the nodules are seen clearly, therefore, it is more likely to show lung tumors than routine chest X-rays. Like CT, Magnetic resonance imaging (MRI) can also provide detailed image of lung. Since the body is not exposed to radiation, there is no known side effects of the MRI. MRI is generally needed to know if cancer has spread to other parts of the body. Positron emission tomography (PET) and bone scans are other types of scans but they are not used widely for diagnosis. Bone scans are used to know if the cancer has spread into any areas of the bones.

In recent years, studies have shown that low-dose CT scans have 20% better performance in early diagnosis of lung cancer compared to traditional chest radiology [24]. This result was found by the National Lung Screening Trial (NLST). NLST, that is a randomized control trial study, conducted a study on 50,000 people who were of age between 55 and 74, and who were current or former smokers. People in the study got either 3 low-dose CT or 3 chests X-ray each a year apart. After several years, the study has found that patients who got low-dose CT have a 20% lower chance of dying as a result of the early diagnosis that can be made on CTs. They were also 7% less likely to die overall (from any cause). The dataset used in this study, which will be explained later in detail, consist of low-dose CT scans.

2.2.2 Automatic Lung Cancer Detection from Medical Images

For the past two decades, automatic detection of the lung nodules has been an active area. Developed computer-aided-detection (CAD) systems were intended to make the nodule detection faster and more accurate. As the new algorithms were tried and processing power has increased, performance of the CAD systems have improved through the years.

Sensitivity and false positives terms are two important parameters that are commonly used in the literature. Sensitivity, which is also called true positive rate, is calculated as the ratio of correctly identified positives to all positives. False positives means number of misclassified sample found as positive but in fact negative.

Armato et al. [3] developed one of the first fully automated computerized methods for the detection of lung nodules from CT scans. Their method was based on 2-dimensional and 3-dimensional analysis of the image data. They applied multiple gray-level thresholds in order to segment the volume. An 18-point connectivity scheme was used to identify contiguous three-dimensional structures. Morphological and gray-level features were calculated for each candidate. A rule-based approach that checks the shape of candidates was applied in order to reduce the false-positives. In the final step, they applied linear discriminant analysis (LDA). The authors reported that this automated method yielded an overall nodule detection sensitivity of 70% with an average of 1.5 false-positives per scan when applied to the 43 cases. When this method was applied to 20 cases, which contained only one or two nodules per case, a corresponding detection sensitivity of 89% with 1.3 false-positives per scan was achieved.

Arimura et al. [4] proposed a scheme based on a difference-image technique for enhancing the lung nodules and suppressing the majority of background normal structures. They obtained the difference image for each computed tomography image by subtracting the nodule-suppressed image from the nodule-enhanced image. The nodule-suppressed image was processed with a ring average filter. Initial candidates were identified by multiple gray-level thresholding techniques. After these processes, a number of false-positives still remained. These false-positives were removed with morphological techniques, gray level thresholding, and ANN model, which was trained for reduction of various types of false positives. This scheme was applied to a confirmed database of 106 low-dose CT scans and provided a sensitivity of 83% for all cancers with 5.8 false-positives per scan. In their dataset, they excluded the nodules that were larger than 30 mm and central nodules, which were endobronchial tumors in or proximal to a segmental bronchus. In this dataset, also, minimum nodule size was 6 mm.

Bae et al. [5] proposed a framework that uses 3D morphological structures. First of all, they applied a gray-level threshold to segment thorax and lung regions, then, they applied 3D region-growing (RG) and labeling. In the final step, they used shape and geometric features by applying 3D multilevel morphological matching in order to

classify the volumes as nodule or non-nodule. Their database consisted of 20 CT scans (13 men, 7 women; age range, 40-75). Slice number per set ranged from 201 to 341, with a mean of 256 images per set. They divided the dataset into two parts: nodules that are 3 mm to less than 5 mm, and 5 mm and larger. They reported a sensitivity of 91.2% for nodules from 3 mm to less than 5 mm, and 97.2% for nodules from 5 mm and larger. The number of false positive detections per patient was 6.9 for 3 mm and larger, and 4.0 for 5 mm and larger.

Belloti et al. [6] presented a CAD system for the selection of nodules in CT images. Their system was based on RG algorithms and a new active contour model (ACM), which were able to draw the correct contour of the lung parenchyma and to include the pleural nodules. As an ACM model, they implemented a local convex hull. Their CAD system consisted of three steps: 1) The lung parenchymal volume was segmented with the RG algorithm. 2) RG algorithm was iteratively applied to the previously segmented volume in order to detect the candidate nodules. 3) Double threshold cut and neural networks were applied to reduce false positives. They reported that detection rate (Sensitivity) of the system is 88.5% with 6.6 false positives per scans for 15 scans that have 26 nodules in total.

Dehmeshki et al. [7] proposed a shape-based genetic algorithm template matching method for nodules with spherical elements. As a pre-processing step, they used a spherical-oriented convolution based filtering scheme for enhancement. 3D geometric shape feature was calculated at each voxel and then combined into a global nodule intensity distribution. Lung nodule phantom images were used as reference images for template matching. Their dataset consisted of 70 CT scans that contain 178 nodules. They reported that 160 nodules were correctly detected by the proposed approach (about 90% detection rate with 14.6 false positives per scan).

Gurcan et al. [8] proposed a framework for detection and classification of the lung nodules. In the first stage, they extracted the lung regions by applying thresholds. This thresholding method successfully separated the surrounding air region and the thorax for all cases. In the thorax region, the histogram of the gray values showed a bimodal distribution. They applied k-means clustering technique with a ratio threshold of R=1. In clustering, only one feature -the pixel gray level value- was used. After extracting

the lung regions, they again applied weighted k-means clustering algorithm in order to detect the nodules. In order to remove normal lung structures (mainly blood vessels), they applied a rule-based system. For example, if the ratio of major and minor axes of a volume exceeded a threshold, it was classified as blood vessels. They also eliminated the *V*-shaped structures because many of the segmented vessels had branching *V*-shapes (Figure 3).

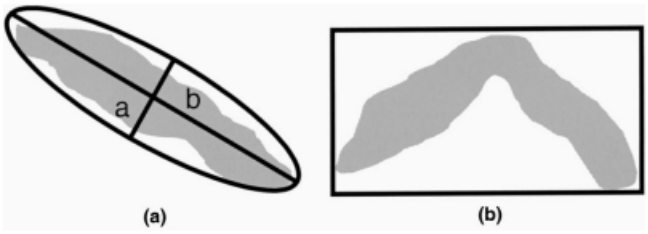

Figure 3: Examples of segmented vessel structures (a) thin long, (b) V shaped [8].

After rule-based classification, a LDA classifier was used to further reduce the number of false positive objects. They reported that the prescreening stage detected 90% (57/63) of the nodules. After false positive reduction using rules based on bounding box, size, and circularity, they obtained a sensitivity of 87% (55/63). The LDA classifier reduced the sensitivity to 84% with 1.74 false positive objects per slice.

Ge et al. [9] proposed improvements to the false positive reduction stage of previous work by using 3D gradient field method and 3D ellipsoid fitting. They formulated 3D gradient field descriptor and derived 19 gradient field features from their statistics. Six ellipsoid features were obtained by computing the lengths and the length ratios of the principal axes of an ellipsoid fitted to a segmented object. Their database consisted of 82 CT scans from 56 patients with the thickness ranging from 1.0 mm to 2.5 mm. They reported a sensitivity of 80% with 0.34 FPs per scan.

Matsumoto et al. [25] proposed a novel filter named quantized convergence index filter (QCI filter) that is capable of enhancing the conspicuity of rounded lesions. QCI filter operated on vector fields consisting of gray-level gradients. Its output at a pixel represents the degree of convergence toward the pixel shown by the directions of gray-level gradients at surrounding pixels. They reported that this degree of convergence is high at any rounded lesion. Therefore, any rounded lesion showing a positive contrast

is detectable by the QCI filter. After they have segmented the lung with gray-level thresholding and 3D connectivity, they applied the QCI filter to the image. Along with the output of the QCI filter, several features (lengths of major and minor axes, effective diameter, elongatedness, contrast etc.) were also used in LDA classifier. They reported that a sensitivity of 90% with 1.67 false positives per slice was achieved, but their database only consisted of 5 cases.

Reported performances of published CAD systems vary substantially because each work uses different datasets and criteria for training and evaluation. For example, dataset size, scan properties, having multiple scans for an individual, slice thickness, and radiation dose were different for each dataset. Moreover, ground truth of the images (state of being nodule in nodule selection criteria) changes because of annotations by different radiologists. Therefore, it is difficult to compare these systems objectively.

ANODE09 was the first study that aimed the evaluation and comparison of the nodule detection algorithms organized by Ginneken et al. [10]. This challenge has allowed research groups to evaluate their algorithms on a dataset that consisted of 55 scans. 5 scans were shared with participants in order to train their algorithms and optimize their internal settings, the remaining 50 scans were retained for testing and not shared publicly. All datasets have been provided by the University Medical Center Utrecht and originated from the NELSON study, the largest CT lung cancer screening trial in Europe. Current and former heavy smokers, mainly men, aged 50-75 years were included in this study. For the scoring system, sensitivity at seven predefined false positives rates, which were 1/8, 1/4, 1/2, 1, 2, 4, 8, were measured. These seven sensitivities were averaged to obtain an overall score of a system. Since the scoring system of the LUNA16 Challenge is the same as ANODE09, this system is explained in the LUNA16 Challenge section in detail. Murphy et al. [26] outperformed all other submissions (5 methods) in this challenge with an average sensitivity of 0.632 (Others: 0.212, 0.291, 0.254, 0.293, and 0.231).

For initial nodule candidates, they used shape index (SI) and curvedness (CV) features [27]. The SI and CV at a voxel were calculated using the principal curvatures k_1 and k_2 at that point:

$$\mathrm{SI} = \frac{2}{\pi}\arctan\left(\frac{k_1 + k_2}{k_1 - k_2}\right) \tag{1}$$

$$\mathrm{CV} = \sqrt{{k_1}^2 + {k_2}^2} \tag{2}$$

Principal curvatures k_1 and k_2 were calculated for all voxels within the lung volume using first and second order derivatives of the image blurred with a Gaussian filter. Voxels which have both SI and CV within the thresholds were selected as seeds. False-positive reduction step consisted of two consecutive classification steps using k-Nearest Neighbor (k-NN) classifiers. They reported that they also tried other classification methods during the experiments and k-NN was found to achieve the best result. The initial classification step used a small number of relatively simple features to quickly reduce most of the obvious incorrect candidates. The second classifier employed more complex features. A total of 135 features were initially considered as being potentially useful. During the experiments, sequential forward floating selection was employed in order to identify most useful features. 8 features were selected for the initial classification and 19 features were chosen for the second classifier.

ANODE09 study has shown that best results were produced by the ensemble of classifiers of all algorithms. It has been shown in the experiments that there was not a single CAD scheme that would be optimal for nodule detection. Different methods have complementary strengths and combining them substantially improves the performance.

ANODE09 study successfully evaluated the nodule detection algorithms but this study only included the 50 scans from a single center, all acquired using one type of scanner and scan protocol. In addition, the ANODE09 set contained a limited number of larger nodules. Therefore, evaluation on a larger and more diverse image database was needed. Several years later, some of the ANODE09 organizers started LUNA16

Challenge, which has more CT scans (888) and a wider variety of nodules in terms of their sizes (3 mm to 34 mm). All methods submitted to LUNA16 Challenge used deep learning techniques and overall performance increased drastically. Although there is not a direct comparison of classical image processing techniques with machine learning and deep learning techniques on the same dataset, when the scores for different datasets were compared, it can be observed that deep learning outperforms the image processing techniques with machine learning. Therefore, LUNA16 will be explained after the Deep Learning section.

Adapting deep learning techniques to medical image analysis domain is a recent trend in biomedical image processing. Why it is a trend and applied in the medical image analysis will be explained in the next section.

2.3 Deep Learning

Deep learning is a new area in the machine learning field. It is based on ANN, which are biologically-inspired computing systems. Today, deep learning algorithms are widely used in image processing, speech recognition, natural language processing, audio recognition, and bioinformatics. In some of these areas, results are comparable or better than the human experts [12]. Deep learning methods have the main advantage over traditional machine learning algorithms like support vector machine or shallow neural networks. The main advantage is that deep learning algorithms extract the features in the data by themselves. Therefore, there is no need for human intervention during the training process. Besides, this feature extraction mechanism generates features that are hard for a human to think and implement. In this section, history and theoretical background are explained in detail.

2.3.1 History

Although deep learning terminology has become famous in recent years, it has a long history that dates back to 1950s. In 1958, Rosenblatt [28] generated one of the earliest types of artificial neural networks, Perceptron. Perceptron, simply, is an algorithm for learning a binary classifier (Figure 4). It is a simple mathematical model of a biological neuron, whose output can be given as:

$$f(x) = \begin{cases} 1, if\ w.x + b > 0, \\ 0, \quad otherwise, \end{cases} \tag{3}$$

where $f(x)$ is the output of the neuron, w is a vector of real-valued weights, $w.x$ is the dot product of the weight vector w and the vector x, and b is the bias. Perceptron is accepted as one of the first artificial neural networks to be produced.

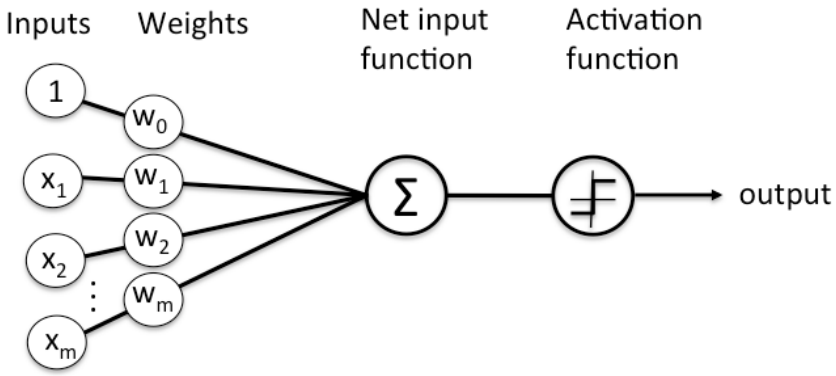

Figure 4: Perceptron algorithm [29].

Although perceptron seemed promising initially, it was understood that this algorithm was very primitive for most of the tasks. Despite being a non-linear function, it separates the input space by a hyper plane. Therefore, if a dataset is not linearly separable, perceptron algorithm cannot converge. For example, it is not able to learn the XOR function.

In 1969, Minsky and Papert published a book called Perceptron [30]. In this book, they addressed two problems related to perceptron. The first problem was that basic perceptrons were incapable of separating the classes when the classes are not linearly separable. The second problem was that computers did not have enough processing power to effectively handle large neural networks. This situation had decreased the interest in neural network field for many years. Although Stephen Grossberg published a series of papers that introduced networks capable of learning XOR functions [31], this field stagnated until the 1980s. In 1980, Fukushima introduced Neocognitron [32], a hierarchical multilayered artificial neural network (ANN), which was an extension to original perceptron. Neocognitron was used in handwritten digit recognition at that time, and performance was quite good. Although interest in neural networks has risen with the Neocognitron, it was still a problem how to train such big models. Although backpropagation algorithm, which is used to train models, was known since 1970s, it has gained popularity after the paper published by Rumelhart et al. [33] in 1986.

In 1989, LeCun et al. applied the standard backpropagation algorithm into multilayer ANN. They used ANN for the purpose of recognizing handwritten ZIP codes on mail. Although the algorithm worked, training time was 3 days, which was impractical [34].

During the late 1980s and early 1990s, many important advances were made by researchers. Bengio et al. identified some of the fundamental mathematical difficulties in neural networks [35]. Hochreiter and Schmidhuber introduced the concept of the long short-term memory (LSTM) network to resolve some of the difficulties [36]. Today LSTM is widely used in many sequence modeling tasks and natural language processing.

In those years, AI companies and ventures on neural networks began to make unrealistic claims in order to get investment. When neural networks did not achieve this goal, investors were disappointed [37]. In the meantime, other fields of machine learning like kernel machines and graphical models started to achieve good results on many important tasks. These two factors caused the decline in the popularity of the ANNs. At that point in time, neural networks were believed to be very difficult to train.

In 2006, Geoffrey Hinton showed a kind of neural network called deep belief network which can be effectively trained using the strategy called the greedy layer-wise pre-

training [38]. This method significantly increased the generalization of the test examples. During this time, ANNs implemented on Graphical Processing Units (GPU) that enabled faster matrix multiplications and parallel programming, and convolutional neural networks (CNN) were introduced. Consequently, researchers trained deeper ANNs which were not possible before. In 2012, Krizhevsky et al. [39] trained a deep convolutional neural network to classify 1.3 million high-resolution images and got a very successful result in the ImageNet Large Scale Visual Recognition Competition (ILSVRC-2012) [40]. This wave of neural networks research popularized the term "deep learning". Nowadays, deep neural networks outperform other AI systems -based on machine learning technologies- on tasks such as image recognition, natural language processing, speech recognition, and audio classification.

Today, deep learning techniques continue to evolve and increase their performance on various tasks. One of the main reasons is that as we use more technology, more data are collected. Since deep learning algorithms are highly dependent on the volume of the data, this situation makes the algorithms perform better. Another reason is that with the increasing CPU and GPU power, more data can be handled at the same time and results are achieved faster compared to the past. This enables more trial and change on the algorithms, resulting in faster convergence to the optimal model.

2.3.2 Neurons

In order to understand ANN's theory and working mechanism, the structure of the biological neuron and biological neural network system must be studied. A neuron is an electrically excitable cell that processes and transmits information through electrical signals (Figure 5). Neurons are connected to each other and form neural networks. A typical neuron is divided into three regions: the soma, dendrites, and axon. Most neurons receive input signals through their dendrites. If the input signal creates an action potential, neuron sends the signal through axon. Many axons are covered with a myelin sheath that helps the fast transmission of the action potential. Axon of the neuron is connected to dendrites of other neurons. Therefore, one neuron can excite other neurons. Information is transmitted in the form of chemical messengers called

neurotransmitters. Neurotransmitter molecules cross the synapse and affect the other neurons.

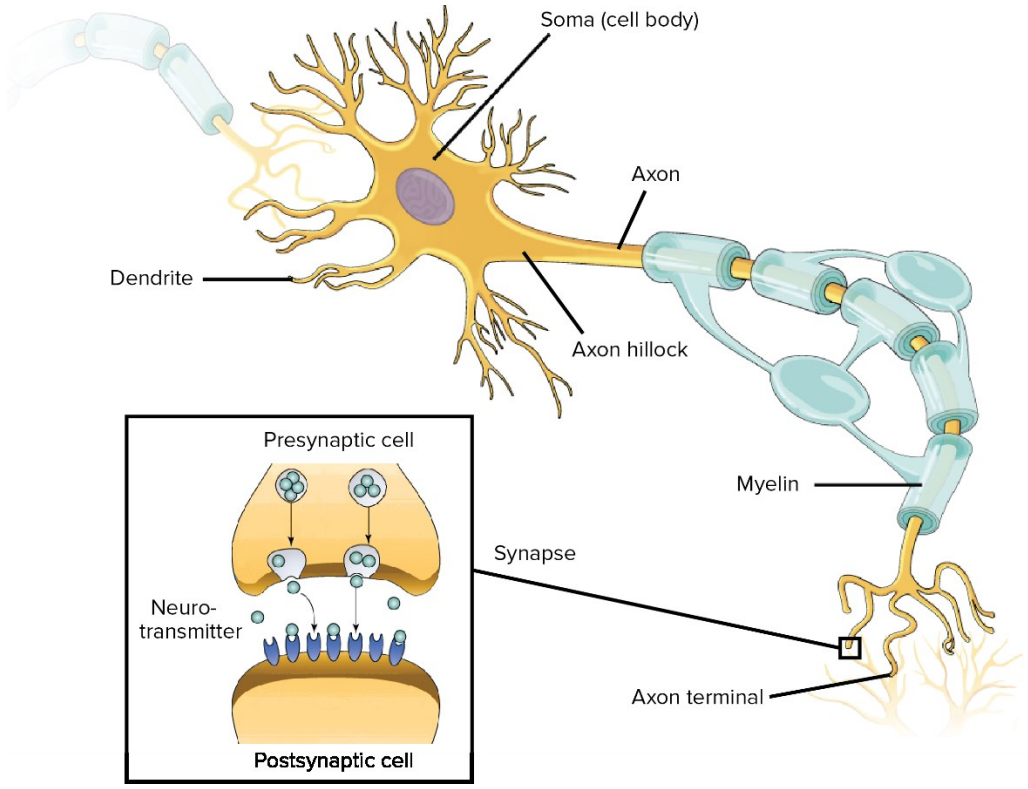

Figure 5: Anatomy of a neuron [41].

When these neurons come together, they form an enormous neural network structure. The adult human brain contains about 100 billion neurons and in average they make 10,000 connections with each other [42]. That is thought to be what makes humans so intelligent. ANNs are the network of artificial neurons, which are very simplified models of biological neurons. That is what makes ANNs so powerful in machine learning field. However, the number of neurons in deep neural network systems are still not comparable to the number of neurons in human. One of the most complex neural network architectures, which is GoogLeNet, has nearly 6.7 million parameters [43].

2.3.3 Multilayer Perceptron

Multilayer perceptron (MLP) is a kind of feedforward ANN. MLP consists of at least three layers of nodes (Figure 6). The first layer is called the input layer and the last layer is called the output layer. Middle layers are called the hidden layers. Due to its hidden layers, MLP can distinguish data that is not linearly separable. In a MLP system, the number of input and output nodes are determined according to the data. For example, in order to design a network architecture for handwritten digit recognition where numbers are stored in 28x28 size images, there will be 784 nodes (one input node for one pixel, 28x28 = 784) in the input layer and 10 nodes (one node for the each number) in the output nodes. Figure 6 shows a simple 3-layer neural network architecture.

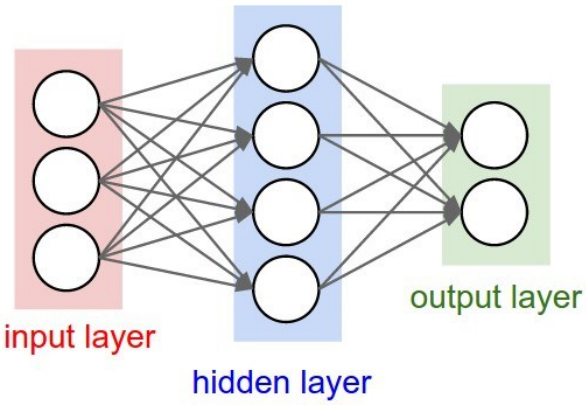

Figure 6: 3-Layer neural network architecture. The input layer has 3 nodes, hidden layer has 4 nodes and output layer has 2 nodes [44].

The number of hidden layers and number of nodes in hidden layers are design issues. In literature, there is no general formula for determining the number of hidden layers. Experiences show that increasing the number of hidden layers can reduce training error but it also increases the complexity of the algorithm and causes a decrease in generalization ability of the system [44]. Besides, it has been observed that as the number of hidden layers increases, the system cannot update the weights and more local minima are observed. On the other hand, if the number of hidden layer and nodes in these layers are not enough for the system, the model does not work properly. As in

the number of hidden layers, determining the number of nodes in the hidden layer is also another design issue. Too many nodes will make training longer and the network may lose its generalization ability. On the contrary, with too few nodes, the network has to use too little information and may not solve the complex models. These hyper-parameters are determined during the training process according to the training results.

In MLP networks, activation functions in nodes are generally chosen as nonlinear functions. Commonly used activation functions are sigmoid (Eq. 4, Figure 7), tanh (Eq. 5, Figure 8), and rectified linear unit (Eq. 6, Figure 9).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

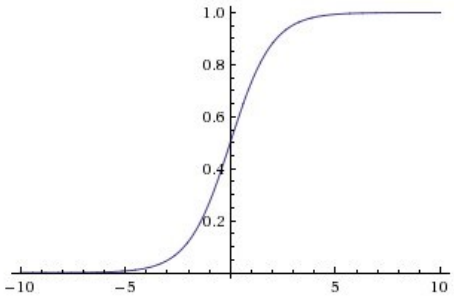

Figure 7: Sigmoid function [44].

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{5}$$

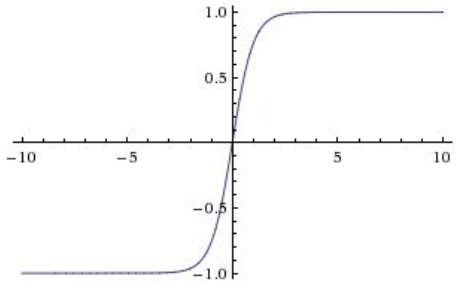

Figure 8: Tanh function [44].

$$f(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases} \quad (6)$$

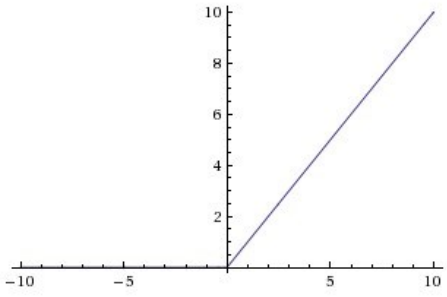

Figure 9: Rectified Linear Unit (ReLU) function [44].

2.3.4 Backpropagation Algorithm

There are two main steps when training MLP with backpropagation. These are the feedforward step and backpropagation.

This training algorithm can be summarized as follows:

1. Initialize the weights,
2. For each sample, apply the feedforward procedure (calculate the output value of the nodes),
3. Use backpropagation to update weights,
4. Repeat steps 2 and 3 respectively until there is a convergence.

Details of the training algorithm are discussed in the Section 2.3.4.1 and 2.3.4.2.

2.3.4.1 Feedforward Step

In the feedforward step, nodes are multiplied with their corresponding weights and results are summed. Resulting summation is processed in a nonlinear activation function. The output of the activation function becomes the value of that node (Figure 10). Starting from the input nodes, this process is applied to all nodes until node values of the output layer are calculated.

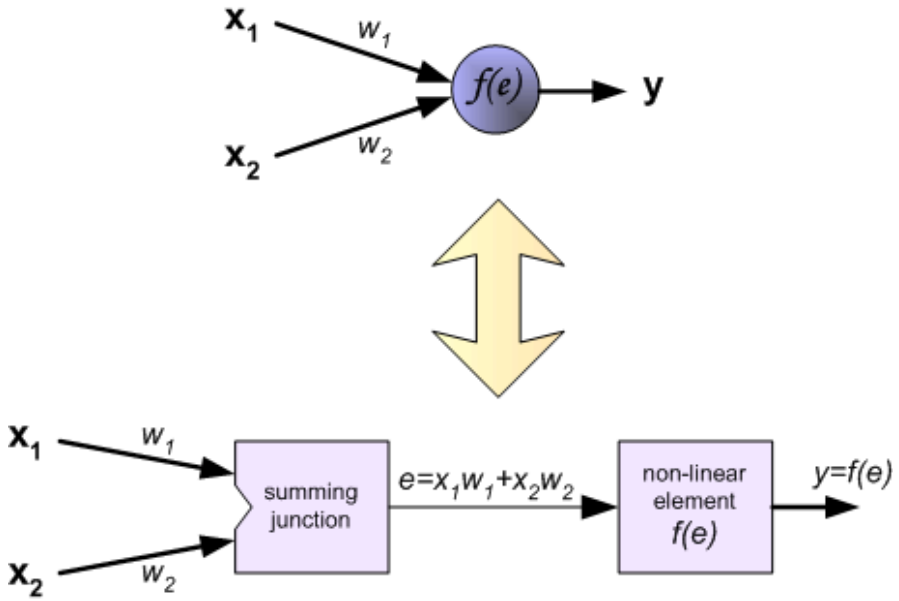

Figure 10: Feedforward process [45].

2.3.4.2 Backward Step

When the feedforward operation has been completed, there is an error between the real output values and the calculated output values. By calculating the difference between the two, error can be easily calculated in the output layer. Yet, error in the inner layers is hard to calculate. If the error in the inner layers cannot be calculated, updating the weights is not possible. For many years, an effective method for calculating the error in inner nodes was unknown.

Backpropagation algorithm has gained its popularity with the paper published by Rumelhart et al. [33]. In this paper, they proposed a new procedure that repeatedly adjusts the weights of the connections in the network in order to minimize the error between the actual output value and the calculated output value. During this process, nodes in the internal layers can learn the certain patterns.

This algorithm mainly looks for how much a specific weight is responsible for the overall error. Error is sometimes referred as loss or cost. In order to get this information, derivative of total error with respect to the weights of the network must be calculated, and weights must be updated according to this information. For the Gradient Descent algorithm, updating the weights is simply achieved by the following formula:

$$w_{ij}{}^{+} = w_{ij} - \alpha \frac{\partial E}{\partial w_{ij}}, \tag{7}$$

where E is the error, w_{ij} is the current weight, α is the learning rate coefficient, and $w_{ij}{}^{+}$ is the new weight. This algorithm updates the weights so that they converge to a point where the error is minimum. The main challenging part here is to calculate the derivative. Consider a three-layer neural network structure where the sigmoid function is used in the nodes, and squared error function is used at the output node:

$$E = \frac{1}{2}(t - y)^2, \tag{8}$$

where E is the squared error, t is the real output of the training sample, and y is the calculated output.

The aim is to find $\frac{\partial E}{\partial w_{ij}}$.

$$\frac{\partial E}{\partial w_{ij}} = o_i \delta_j \tag{9}$$

$$\delta_j = \frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial net_j} \tag{10}$$

$$\delta_j = \begin{cases} (o_j - t_j)o_j(1 - o_j) & if\ j\ is\ an\ output\ neuron, \\ \left(\sum_{l \epsilon L} \delta_l\, w_{jl}\right) o_j(1 - o_j) & if\ j\ is\ an\ inner\ neuron, \end{cases} \tag{11}$$

where o_i and o_j are output, and net_j is the input value of the neurons.

Figure 11 shows the update procedure of an output weight. The main logic is to obtain the derivative of the error with respect to the weight by using the chain rule.

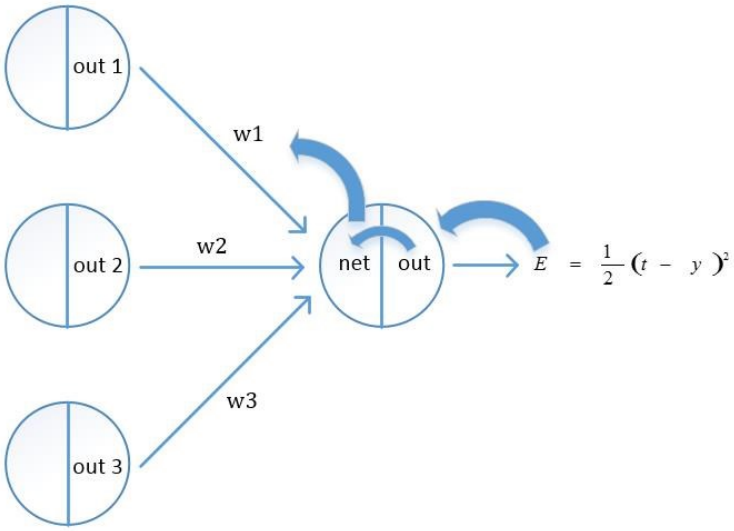

Figure 11: Update procedure of an output weight. Arrows show the derivative steps.

2.3.4.3 Best Practices in Multilayer Perceptron

Backpropagation algorithm does not guarantee convergence to the global minimum. During the training process, there might be various local minima and training process can be stuck in these areas.

One way to optimize convergence is using the adaptive learning rate. In order to avoid oscillations, the learning rate is started with a high value and through the process, it is

decreased gradually. This is one of the simplest methods in adaptive learning rate techniques and this kind of algorithm may improve the convergence rate.

Another widely used and more advanced method is using the inertia. This method is inspired by the idea of a ball rolling down a mountain. Ball's current speed is determined also by its own momentum. For example, a ball rolling down a mountain can pass through local minima because its speed does not depend only the slope of the point where it is located. Even if there is a hill in front of the ball, it can pass through due to its speed.

Momentum term can be added to the learning process as in the Eq. 12.

$$\Delta w_{ij}(t) = -\alpha \frac{\partial E}{\partial w_{ij}} + \gamma \Delta w_{ij}(t-1), \tag{12}$$

where γ is the momentum rate and $\Delta w_{ij}(t-1)$ is the previous change.

With the momentum term, probability of being stuck in a local minimum is decreased.

AdaGrad (Adaptive Gradient Algorithm) is an adaptive learning rate method proposed by Duchi et al [46]. This strategy generally improves convergence over the standard gradient descent algorithm.

The derivative for weight w_{ij} at iteration *t* is defined as follows:

$$g_{t,ij} = \frac{\partial E}{\partial w_{t,ij}} \tag{13}$$

AdaGrad modifies the learning rate coefficient α with the G coefficient:

$$G = \sum_{t=1}^{T} g_{t,ij} g_{t,ij} \tag{14}$$

$$w_{ij}{}^{+} = w_{ij} - \frac{\alpha}{\sqrt{G}} g_{t,ij} \tag{15}$$

G is the sum of the squares of the gradients with respect to w_{ij} up to time step *t*. AdaGrad's main benefit is that it eliminates the need for tuning the learning rate manually. AdaGrad adapts updates to each weight to perform larger or smaller updates according to their importance. AdaGrad's main weakness is that accumulated squared gradients keep growing during the training. This situation causes learning rate to decrease so that algorithm is no longer able to converge to a point.

RMSProp is a very effective method which is an update to AdaGrad. RMSProp is an unpublished adaptive learning rate method which is mentioned in Geoffrey Hinton's Coursera class [47]. RMSProp adjusts the AdaGrad method so that learning rate does not decrease monotonically. The idea is to divide the learning rate constant by an exponentially decaying average of squared gradients.

$$M(t) = \vartheta M(t-1) + (1-\vartheta)G, \tag{16}$$

$$w_{ij}{}^{+} = w_{ij} - \frac{\alpha}{\sqrt{M(t)}}g, \tag{17}$$

where ϑ is the decay rate. Adam [48] is a recently proposed adaptive learning algorithm method. This algorithm combines both RMSProp and momentum approach.

$$m(t) = \beta_1 m(t-1) + (1-\beta_1)\frac{\partial E}{\partial w_{ij}}, \tag{18}$$

$$v(t) = \beta_2 v(t-1) + (1-\beta_2)\left(\frac{\partial E}{\partial w_{ij}}\right)^2, \tag{19}$$

$$w_{ij}{}^{+} = w_{ij} - \frac{\alpha}{\sqrt{v(t)}}m(t), \tag{20}$$

where β_1 and β_2 are the learning hyper-parameters and recommended values in the paper are 0.9 and 0.999 respectively. Today, Adam is the most popular and widely used adaptive learning algorithm.

When training the neural networks model, there are two modes of learning, which are stochastic and batch learning. In stochastic learning, only one sample is propagated and it is followed by backpropagation. Therefore, there is a weight update in each process. On the other hand, in batch learning, all of the dataset is propagated and weights are updated according to the cumulative sum of all dataset. Stochastic gradient descent is observed to work better if there are many local minimums and maximums. In contrast, batch learning yields a faster convergence to the local minimum points [49]. Their usage depends on the dataset used in the training. In general, mini-batch is used in modern applications where the batch is a randomly selected small sample of all dataset.

Choosing the learning rate constant is another important design issue. With the low learning rates, improvements will be linear. On the other hand, high learning rates will decay the loss function faster but they might be stuck in local areas. By analyzing loss graphs in the training process, an optimum value can be decided. A high learning rate generally decreases rapidly and remains constant after a while. On the other hand, a low learning rate may not reach the global minimum due to local minima, too. Typical behaviors of the learning rates are shown in Figure 12, where loss refers to the error and epoch refers to the number of iterations.

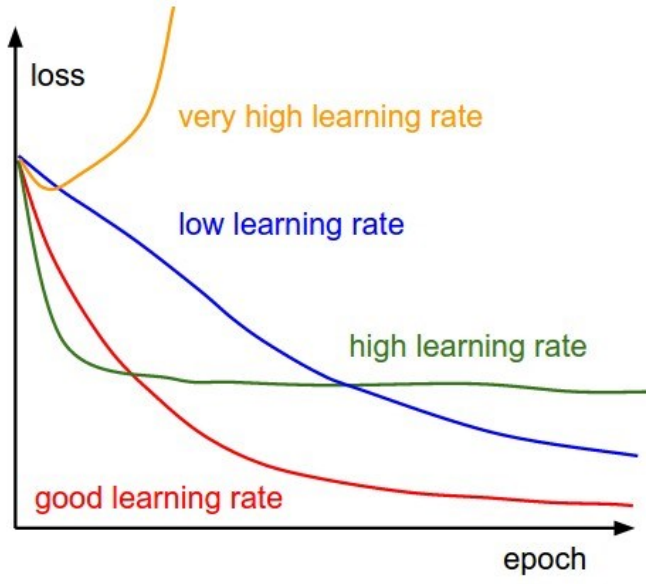

Figure 12: Different learning rates and their possible performances [44].

When training the neural networks, if few epochs are used, the model might underfit (cannot learn enough, both test and training set accuracies are low); if too many epochs are used, the model might overfit (model memorizes the training samples). Overfit means that the test set has low accuracy although the training set accuracy is very good. In the context of neural networks, accuracy refers to the ratio of the number of correctly classified samples to the all samples.

In order to overcome this situation, accuracy of the training set with different epochs, and result of the test set for those epochs must be analyzed. This analysis can show the user when to stop the training process (Figure 13). Since the model is trained on the training set, there will always be a slight difference between the training set accuracy and the test set accuracy. When this difference increases through the training, it is a sign for terminating the training process.

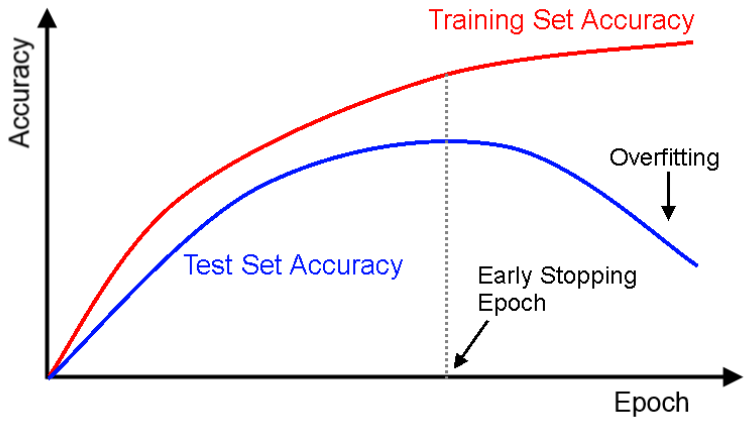

Figure 13: Relation between test and training accuracy with increasing epoch [50].

It is observed that neural networks learn more quickly and give better performance if the data are preprocessed before the training.

Mean subtraction is one of the most common forms of preprocessing. By subtracting the mean from every element, data are gathered around the origin along each dimension.

Normalization refers to the normalizing the data dimension so that they are on the same scale. One way of normalization is to divide each dimension by its standard deviation.

Another way of normalizing the data is finding the minimum and maximum values of the data and rescaling the data to be between -1 and 1.

When initializing the neural networks, values of the weights must also be initialized to some values. Setting all weights to zero makes all neuron outputs the same value. This results in computing the same gradients during the backpropagation and yields the same parameter updates. Therefore, there should not be any symmetry in the values of weights. Generally, weights that are very close to zero but not identically zero are preferred in the initialization of the network [44]. For example, initializing the weights randomly with zero mean and unit standard deviation Gaussian is a preferred way of initialization. Yet, small weights are not always proper for every neural network. With small weights, the update might be very small because gradient is directly proportional to the weight. In order to spot this issue, accuracy and loss graphs must be analyzed in detail when training the network.

In order to prevent overfitting during the training process, dropout technique can be used. This technique, which is extremely effective and simple, was introduced by Srivastava et al. [51]. Dropout is a technique where randomly selected neurons are ignored during the training. In order to apply the dropout procedure, a hyper-parameter p ($0 \leq p \leq 1$) is determined. At each training stage, individual nodes are kept with probability p. Only the remaining nodes are used during the training of that step (Figure 14). After the training step, the nodes ignored are inserted back into the network with their original weights multiplied by p.

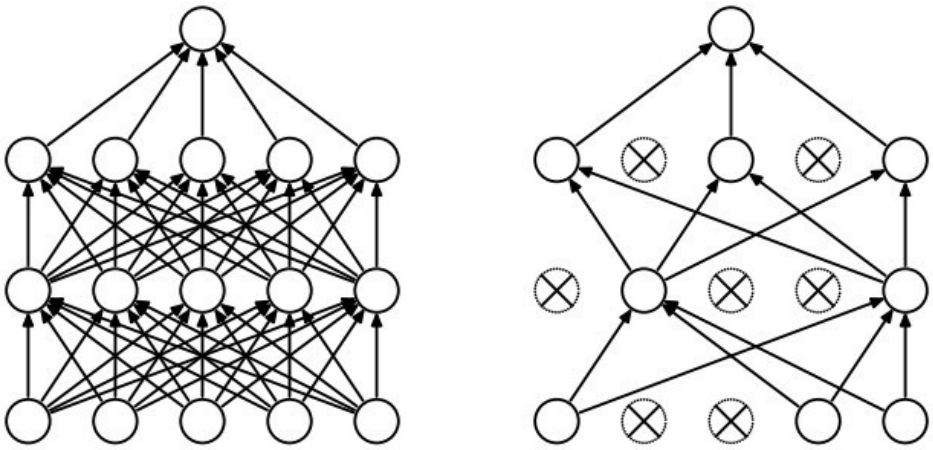

Figure 14: Right net is the result of applying dropout. Crossed nodes have been dropped [51].

Averaging the results of different architectures generally improves the performance because it removes the noises coming from different models. Yet, training many separate networks is computationally expensive. Dropout provides almost different architectures for same model. Besides, neighbor neurons can also memorize the same patterns. If this situation goes too far, the model will be very fragile, which means being too specialized to the training data. Yet, we want model neurons to learn different patterns so that each of them is an independent feature detector. This phenomenon that occurs during training is referred to complex co-adaptation [51]. Dropout mechanism causes the network to become less sensitive to specific patterns. This results in a network that is capable of better generalization and less likely to overfit training data.

2.3.5 Convolutional Neural Networks

Convolutional neural network (CNN, or ConvNet) is a class of feed-forward ANN. CNNs use very little pre-processing compared to regular neural network models. Network model in CNN learns the features itself because it builds the relationships between adjacent nodes. For example, when classifying hand-written digits (28x28 pixel), MLP gets each pixel individually and forms a vector whose size is 784. Yet, it

misses the information between adjacent pixels. Each pixel has a spatial relationship with its neighboring pixels. However by vectorising the input, this information is lost. It is also the same for natural language processing. Each word or syllable depends on its previous or next word or syllable. Yet, MLPs discard this spatial information and process each input node independently. In order to overcome this situation, generally, pre-processing step is applied.

In CNN, this spatial information is taken into account by a convolution step. Filters learn their values automatically during the training and reveal specific patterns in the data. CNNs are generally composed of convolution layers, pooling layers, and fully connected layers. In brief, CNNs extract the specific patterns by using the filters, then pooling layers help the model ignore redundant data. By applying convolutions and pooling respectively, only certain patterns remain. As a final step, resulting data is vectorized and MLP is used in the last step.

In CNN, raw data is represented as tensor. Tensor concept can be generalized as higher order matrices. For example, a vector is an order 1 tensor, gray-scale image is an order 2 tensor, and an image that has three channels (R, G, and B) is order 3 tensor. The input, intermediate representations, and parameters are all represented as tensors in CNN model.

2.3.5.1 Convolution Layer

The primary function of a convolution layer is to extract features from the input data. Convolution is a mathematical operation of two functions. In the CNN concept, convolution operation simply slides a kernel function, which is also called filter, over the main data by performing element-wise multiplication of each element. For each window in the sliding process, the sum of the element-wise multiplication gives the result for that window. By sliding windows through the whole image, the output of convolution operation, which is called feature map, is produced.

In Figure 15, a matrix of size 5x5 is convolved with a kernel of size 3x3. Starting from the upper left corner, kernel slides through the whole image. Convolved feature in the

last image shows the feature map. It can be observed that when there is a correlation between the kernel and the input image, resulting feature map has higher values in those areas. When there is not any resemblance, resulting feature map has lower values in those regions. That is why kernels are also referred to as the “feature detectors”. In the convolution layer, many kernels are used on the original data. Each of these kernels learns different patterns and features of the input data. For example, they can learn to detect the edges, curves, blobs, and smooth areas.

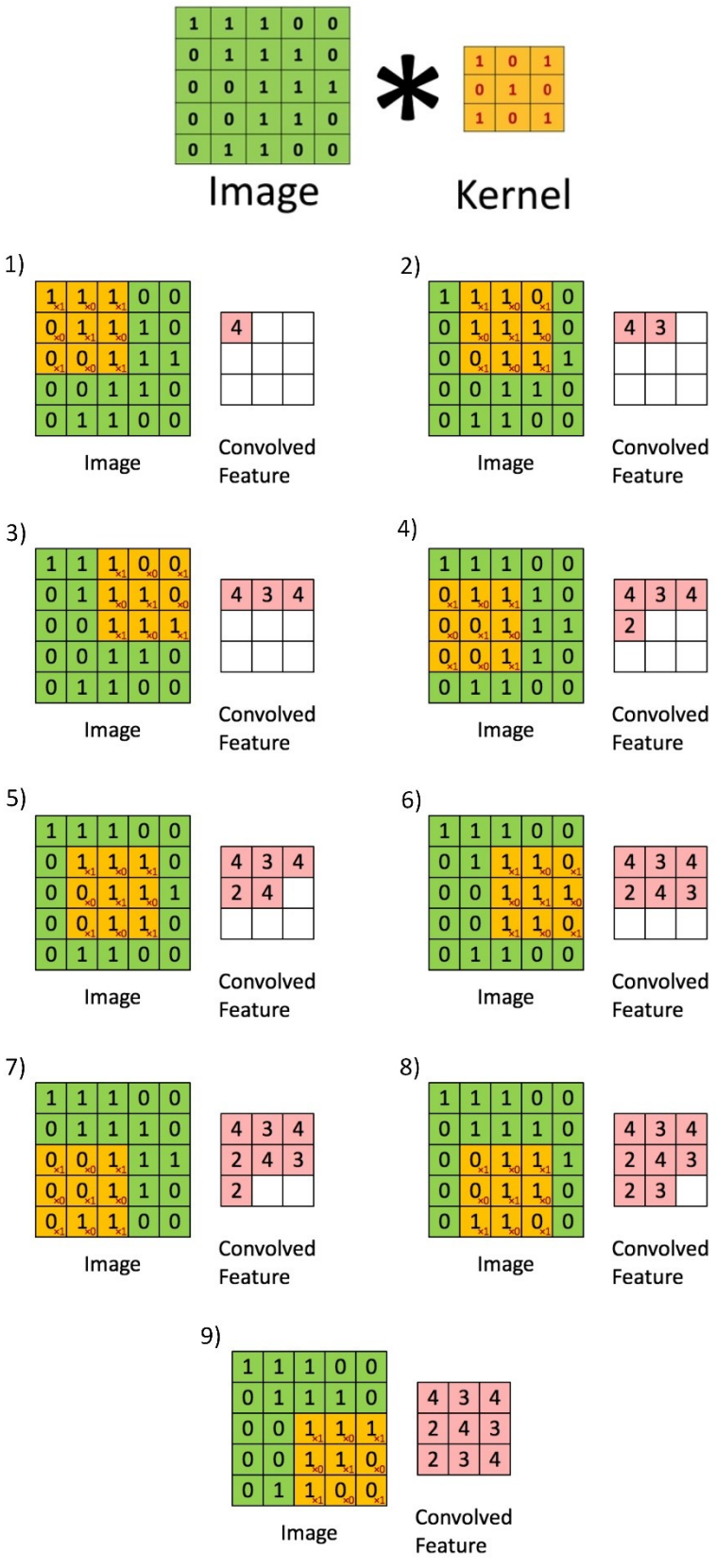

Figure 15: Convolution operation [52].

When considering the convolution operation in CNNs, there are three design issues to think about: kernel size, number of kernels, and stride. These hyper-parameters have an effect on the shape of the output data and memory usage.

Kernel size determines the receptive field of each neuron at the output of the convolution layer. Values of the kernel represent the weights of the model. For example, if the kernel has size 3x3, its output has a connection with 9 different nodes in the input layer. Generally, kernel dimension is the same as the input data dimension. If the input is a 1 channel greyscale image, the kernel is also 1 channel. If the input is a 3 channel RGB image, the kernel is selected as a 3-dimensional structure. If the kernel size is very small (eg. 2x2), it will not be able to extract enough features. For example, small kernels cannot detect big complex patterns. Yet, if convolutions with small kernel sizes are applied consecutively, they can extract features. On the other hand, bigger kernel size increases computation complexity. Generally, small kernel sizes such as 3x3 or 5x5 are used in the CNN training. However, recent publications also state that 1x1 filters may be useful in some cases in order to reduce the dimension in the network [43].

Number of kernels is a very important design parameter because it determines the number of different features to be focused on. If the number of kernels is small, the network may miss some of the patterns in the data. On the other hand, the number of kernels should not be big enough to cause duplicate filters. For example, if the kernel size is 3x3, using 64 kernels will create duplicate filters because 3x3 size cannot create 64 qualified different filters. In addition to duplicate filters, too many filters will bring memory issues because each convolved image takes up space in the memory of the computer.

When each kernel is slid over the input image, they generate a feature map. These output images are concatenated after all kernels have produced their outputs. If the input image is 2-dimensional, its outputs will be 3-dimensional tensor. If the input image is 3-dimensional volumetric data, its outputs will be 4-dimensional tensor. This extra dimension comes from using many filters. Depth of the tensor and depth of the kernel must be same. For example, consider an input tensor of a size 30x30, and a kernel of a size 3x3. The resulting size of the convolution is 28x28. Yet, there are more

than one kernel, which are applied on the input tensor. As a result, size of the output tensor becomes Kx28x28. For the next convolution operation, kernel dimension becomes KxNxM. In CNN model representations, this size is generally represented as K@NxM.

Striding operation controls how the filter convolves around the input volume. Generally, kernel slides through input image 1 node step size at a time. Yet, in order to control output tensor size, stride number can be changed. If it is changed to 2, the kernel will go to next slide by sliding 2 nodes. For example, consider an image of size 64x64 and a kernel of size 3x3; if the stride is 1, the output image will be 62x62. If the stride is 2, the output image will be 31x31.

In convolution operation, padding can be added to the input tensor, in order to keep the size of the output tensor as same as the input volume. Padding size is generally determined according to the size of the kernel. For example, if the kernel size is 3x3, 1 pixel around the image is sufficient. If the kernel size is 5x5, 2 pixels around the image can be used (Figure 16). Padding values are generally selected as zero or the same value as the edge pixels.

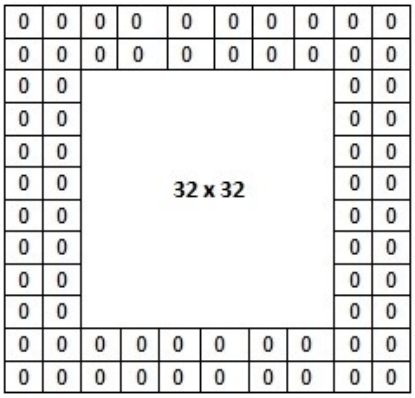

Figure 16: For a size of 5x5 kernel, two pixels zero padding can be used in order to keep the image size the same.

The equation for calculating the output size for any given convolutional layer is:

$$O = \frac{(W - K + 2P)}{S} + 1, \tag{21}$$

where O is the output height/length, W is the input height/length, K is the filter size, P is the padding, and S is the stride.

Figure 17 shows the filters after the training of a CNN created by Krizhevsky et al. [39]. Shapes and colors in the filters were formed iteratively during the training process.

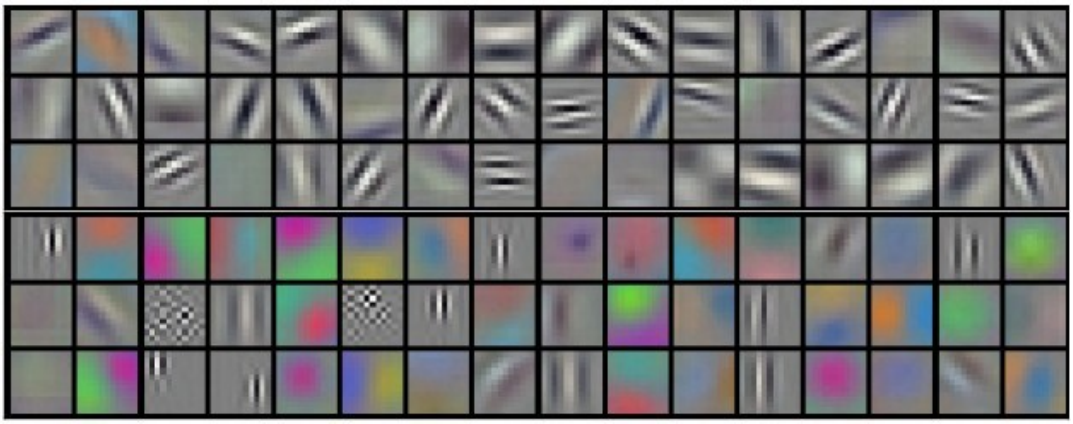

Figure 17: Example filters learned by Krizhevsky et al. [39], filter size is 11x11x3 pixel.

2.3.5.2 Pooling Layer

Pooling layer, which is also referred to as the downsampling layer, reduces the spatial size of the output of the convolution layer in order to reduce the number of parameters and computation in the network. Pooling layer is also used for controlling overfitting. Pooling layer is generally settled between two convolution layers or convolution and fully connected layers.

Max pooling and average pooling are two of the mostly used pooling methods. For a given window, max pooling takes the maximum value in that window and average pooling takes the average value of the values in the window. For pooling operation, there are two important hyper-parameters which are window size and stride value.

Window size determines the width of the area to be focused on. On the other hand, stride determines the step size of the sliding window. For example, if the input tensor size is 8@32x32 and the stride is 2, output tensor will be 8@16x16. In Figure 18, max pooling is illustrated.

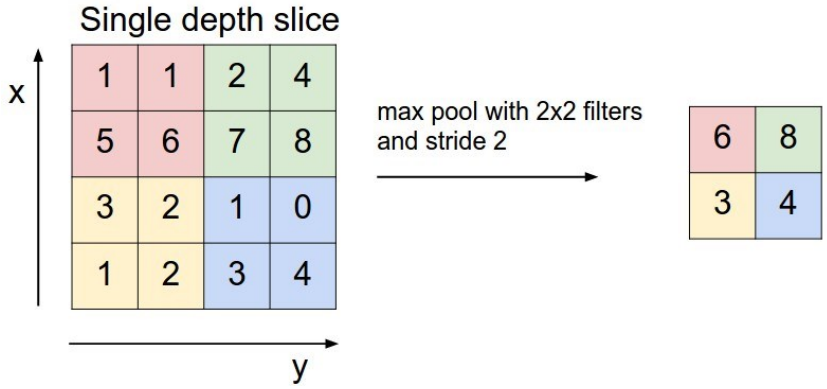

Figure 18: Max pooling [44].

2.3.5.3 Fully Connected Layer

Convolution and pooling layer generate rectangular shaped outputs. These outputs are converted to vector format so that they can be multiplied by the weight matrix. For example, if there are 64 feature map layers each of which has 5x5x3 voxels, in the fully connected layer these volumes are converted to a 4800x1 vector (5x5x3x64 = 4800). The layer before the fully connected layer represents high-level features. With the help of a fully connected layer, these high-level features can be multiplied by the weights of the hidden layers. Rest of the system works as MLP do.

2.3.5.4 CNN Patterns

Generally, a convolutional layer followed by a rectified linear unit (ReLU) and pooling layer is used multiple times before the fully connected layer. In CNN architecture, earlier convolution operations look for low level features such as lines and edges. On the other hand, later convolution operations try to look for high level features, which are specific to the training data. If there are more convolutional layers in the network,

model tries to look for more high level features. Some mostly used CNN patterns can be given as follows:

$Data \rightarrow [C \rightarrow ReLU \rightarrow P]^N \rightarrow FC$

$Data \rightarrow [C \rightarrow C \rightarrow P \rightarrow C \rightarrow C \rightarrow P]^N \rightarrow FC$

$Data \rightarrow [[C \rightarrow ReLU]^N \rightarrow P]^M \rightarrow FC$

where C is the convolution layer, ReLu is the rectified linear unit, P is the pooling layer, and FC is the fully connected layer. N and M parameters stand for how many times the operation group applied consecutively.

CNNs are very popular in image processing area, and in recent years they provided state of the art performance in image recognition tasks. LeCun et al. [53] proposed a CNN model that was designed for handwritten digit recognition. This model was trained with 32x32 pixel images and consisted of 7 layers (Figure 19).

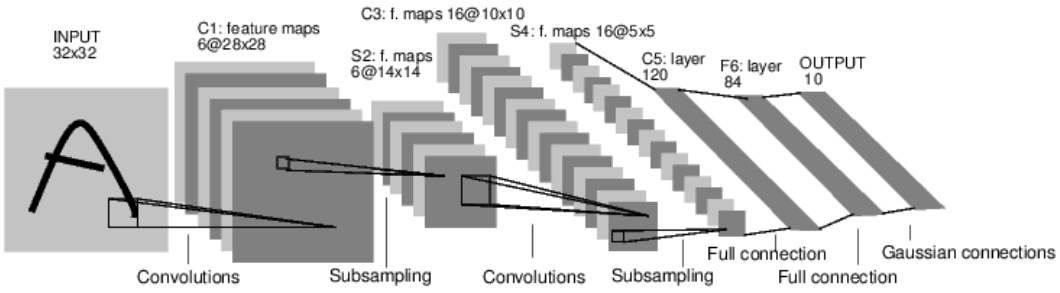

Figure 19: LeNet-5 architecture [53].

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a benchmark in object classification and detection tasks [54]. Since 2010, it is organized annually and today there are nearly 14 million images in their database. In recent years, most of the models submitted to the challenge are CNN models. In 2012, Krizhevsky et al., [39] proposed a CNN model called AlexNet which outperformed the other models. AlexNet achieved a top 5 error rate of 15.4%. That year, next best entry achieved an error of 26.2%. This result was very good compared to the other submissions. Therefore, CNNs have started to be used widely in computer vision tasks then on.

Compared to today's CNN models, AlexNet has a simple model consisting of 8 layers (Figure 20).

Since training took too much time, problem set was divided into two parts which are trained on separate GPUs.

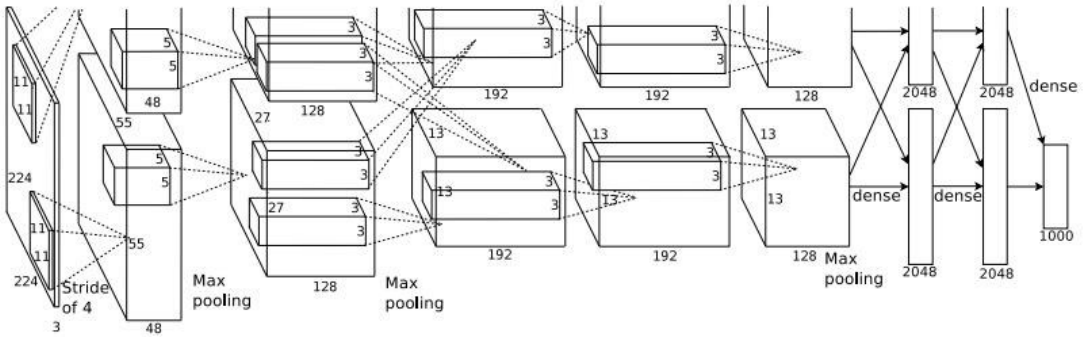

Figure 20: AlexNet architecture. First 5 layers are convolutional and last 3 layers are fully connected layers [39].

In 2014 GoogLeNet [43] architecture won ILSVRC-14 by achieving a state of the art performance (Figure 22). GoogLeNet model consists of 22 layers. This model achieved top 5 error rate of 6.67% [55]. In their model, Google introduced the concept of inception (Figure 21).

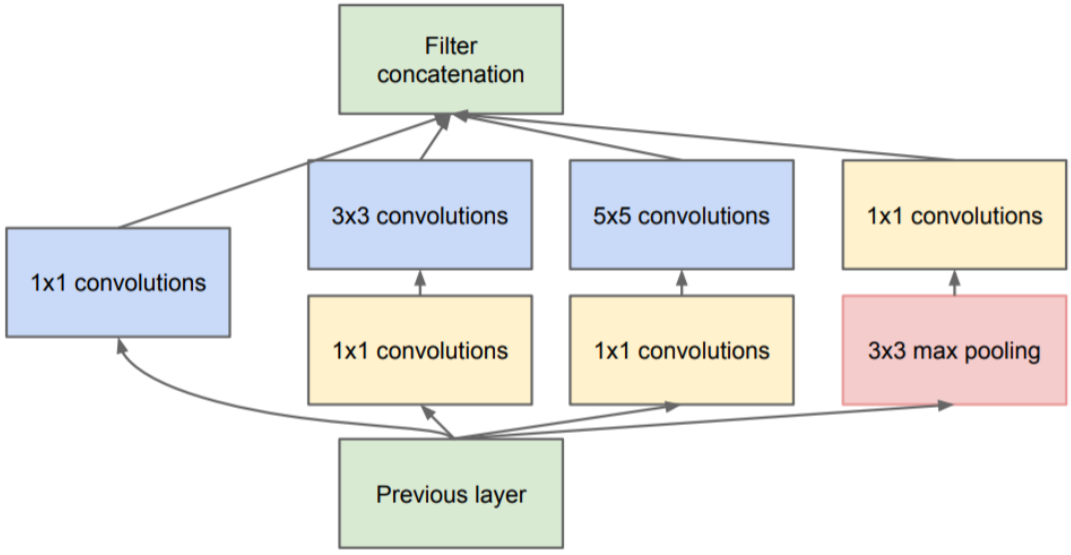

Figure 21: Inception module.

In inception module, there is a medium sized filter convolution, a large-sized filter convolution, and a pooling operation. This network in network structure is able to extract information about the very fine details and models in the volume. Besides, it reduces the number of parameters used in the network. It uses 12x fewer parameters than AlexNet. ReLU is used after each CNN layer in order to keep non-linearity of the network.

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figure 22: GoogLeNet architecture [43].

2.4 LUNA16 Challenge

In this challenge [15], a large dataset containing 888 CT scans with annotations from publicly available LIDC-IDRI database was used for both training and testing. This database is available from NCI's Cancer Imaging Archive [56] under a Creative Commons Attribution 3.0 Unsupported License. The LIDC-IDRI database contains a total of 1018 CT scans but challenge organizers discarded the scans that have a slice thickness more than 3 mm. In addition, they removed the scans with inconsistent slice spacing or missing slices. After removals, 888 scans remained. These scans were provided in MetaImage format (.mhd) and each .mhd file was stored with a separate .raw file that stores pixel data. Dataset can be accessed from the LUNA16 website [15]. In Figure 23, three different profiles of a CT scan from the dataset can be seen. The upper left profile is called as the transverse plane, the upper right profile is called as sagittal plane, and the lower right profile is called as the coronal plane. When the 2 dimensional images are used in the algorithms, generally, transverse plane is used.

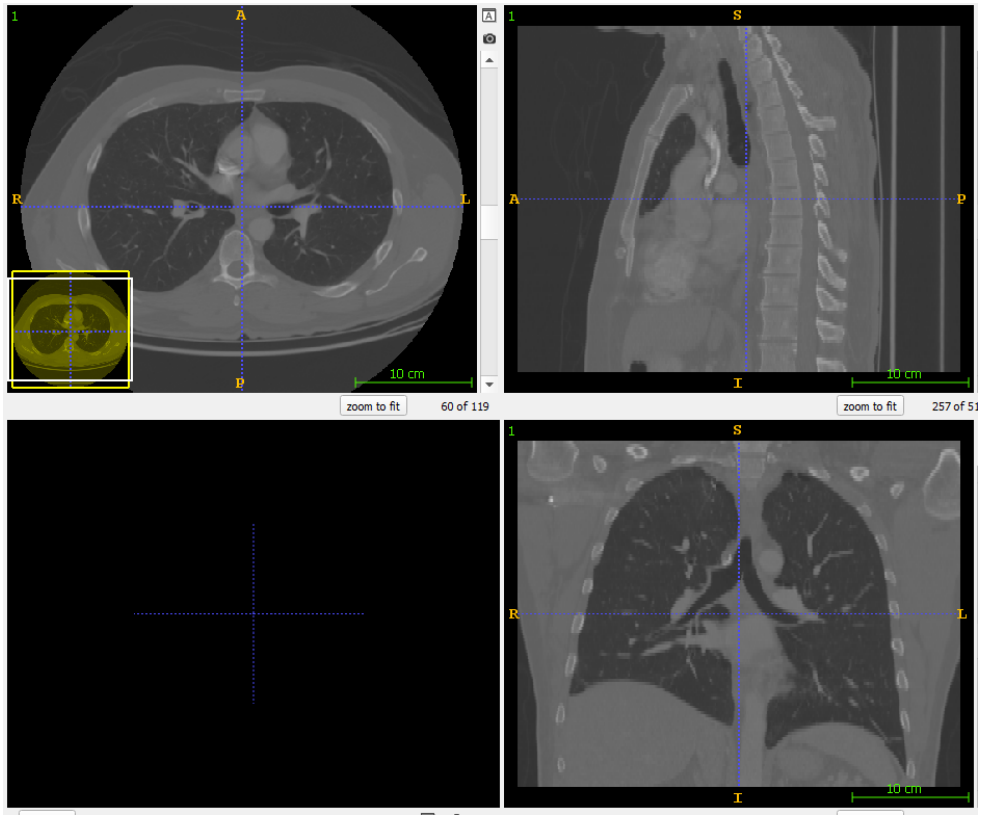

Figure 23: Typical .mhd file opened in dicom viewer software (MiroDicom).

Each LIDC-IDRI scan was annotated by experienced thoracic radiologists in a two-phase reading process [57]. In the first phase, four radiologists annotated the scans independently. All lesions were marked as nodule ≥ 3 mm; nodule < 3 mm; non-nodule. In the second phase, independently annotated nodules were revealed to each radiologist, who reviewed all of the marks again. Organizers only considered the annotations that were categorized as nodules ≥ 3 mm. This resulted in a set of 2,290, 1,602, 1,186 and 777 nodules annotated by at least 1, 2, 3, or 4 radiologists, respectively. They considered the 1186 nodules annotated by the majority of the radiologists (at least 3 out of 4 radiologists) as positive examples in reference standard. In Figure 24, some of the true positives can be seen. As it is seen in the figure, their shapes can be very different from each other. While some of them has a solid rounded shape, others may have more fringed structures.

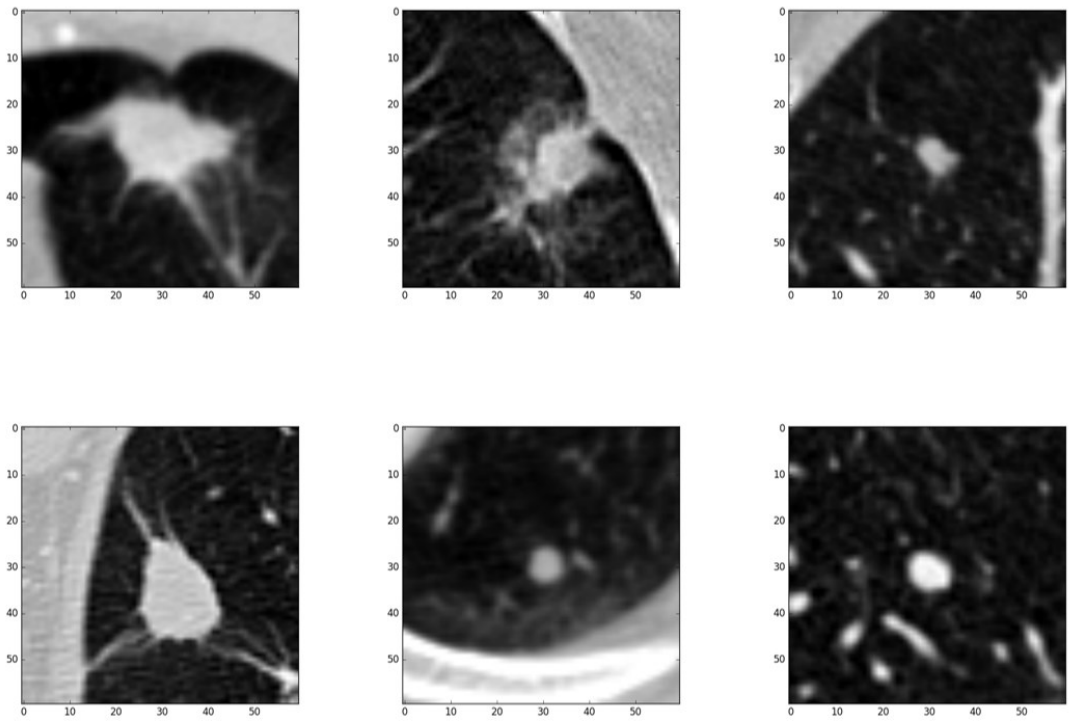

Figure 24: Example true positives.

Challenge consisted of two separate tracks:

1. Complete nodule detection
2. False-positive reduction

The complete nodule detection track required the participants to develop a complete CAD system. Input into the system was only the CT scans. In order to train the algorithms, 1186 nodules locations with their diameters were also provided.

For the false-positive reduction stage, participants were required to classify a number of locations in each scan as being nodule or not. A list of nodule candidates, which were computed using existing nodule detection algorithms, were supplied to the participants. This list was generated by merging the candidates that were detected by Murphy et al. [26], Jacobs et al. [58], Setio et al. [59], Tan et al. [60], and Torres et al. [61]. In total, there were 551,066 candidate points.

Participants were required to perform the 10-fold cross-validation. The dataset has been randomly divided into ten subsets of similar size. The following steps describe how to perform 10-fold cross validation for fold n (n=1, 2 … 10):

1. Use the subset n as the test set and the remaining 9 sets as the training set.
2. Train the algorithm on the training set.
3. Test the training algorithm on the test set and generate the result file.
4. After iterating this process for all folds, merge the result files to get the result for all cases.

For the evaluation of algorithms, participants submitted a .csv file that includes all the candidate points with their x, y, and z coordinates, and their probabilities for being a nodule. In order to assign a class (nodule or non-nodule) to the probabilities, a threshold value must be determined. In order to determine this threshold value, FROC plots are generally used. FROC is a graphical plot that is drawn with sensitivity on y-axis and false positive rate per scan on the x-axis. This curve is drawn by using different threshold values of the binary classifier. An algorithm checks each threshold value and records its false positive rate and sensitivity. Since there can be different threshold values for a certain false positive rate per scan, there can be different sensitivity levels for the same false positive per scan. That is why there are dashed lines in the FROC graph; those dashed lines show maximum and minimum sensitivity values (Figure 25).

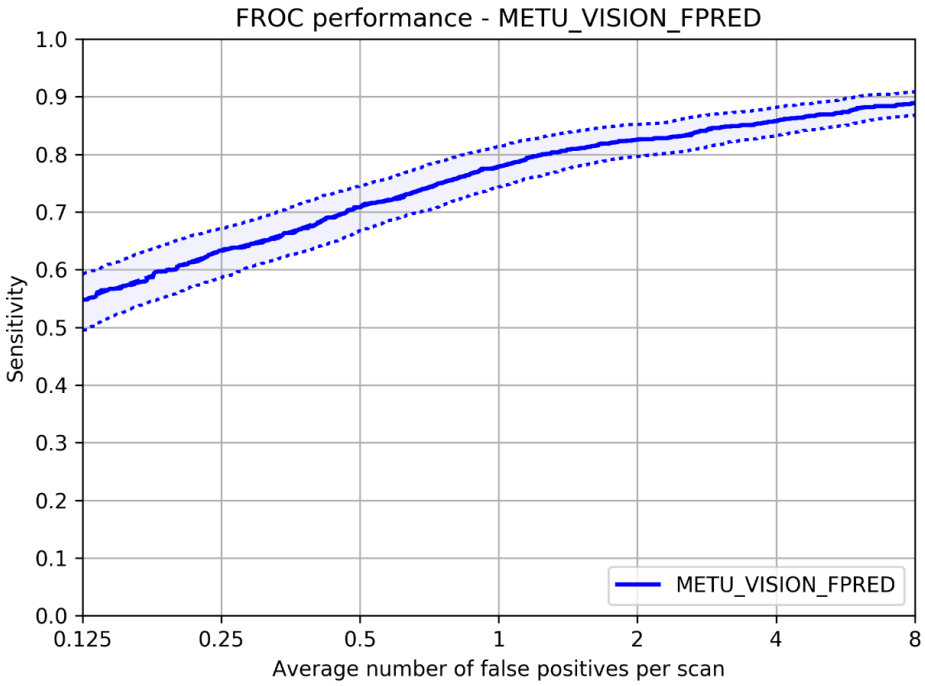

Figure 25: Sample FROC curve.

High sensitivity with low false positive rate is the aimed situation. Yet, there is a tradeoff between sensitivity and false positives. If a low threshold value is set, sensitivity will be high but false positives will be also high. This is not a desired situation because many false positives need to be cleaned out by the radiologists. On the other hand, if a high threshold value is set, false positives will be low but sensitivity will be also low. This is also not a desired situation because there will be many missing true positives, which is not applicable to the clinical use. FROC plot is a good tool to see overall picture and determine the threshold value.

In order to calculate the score of the algorithm in LUNA16 Challenge, sensitivity values at seven predefined false positive rates (1/8, 1/4, 1/2, 1, 2, 4, and 8) were averaged.

Team ZNET [16] used CNNs for both candidate detection and false positive reduction steps. For the candidate detection step, they used the probability map given by U-Net [62]. U-Net was applied to each axial slice. The candidates were extracted based on the slice-based probability map output of the U-Net. After applying thresholding, morphological operations and connected component analysis, they generated the

candidates. For the false-positive reduction step, they used the recently published wide residual networks [63]. For each candidate, 64x64 patches from the axial, sagittal and coronal views were extracted and each of them were processed separately by the wide residual networks. Ensemble of classifiers was used in the end by taking the mean values of the predicted output values of the network.

Xavier initialization [64] was used for weight initialization and ADAM [48] was used as the optimization method. Leaky rectified linear units were used as nonlinearities throughout the network. For the complete nodule detection system their score was 0.811, and for the false-positive reduction step their score was 0.758.

ETROCAD is a CAD system adapted from Tan et al. [60]. For the nodule detection part, they applied resampling to get a voxel dimension of 1 mm. They applied a nodule segmentation method based on nodule and vessel enhancement filters and a computed divergence feature to locate the centers of the nodule clusters. Thresholding on the filtered image and divergence of the normalized gradient was applied to obtain the list of candidates. For the false-positive reduction step, they computed a set of features for each candidate, including invariant features defined on 3D gauge coordinates system, shape features, and regional features. The classification was performed using an SVM classifier. They only participated in complete nodule detection system and got a score of 0.676.

M5LCAD is a CAD system developed by Torres et al. [61]. For the candidate detection part, they used two different algorithms: LungCAM and Voxel-Based Neural Approach (VBNA). LungCAM is inspired by the life-cycle of ant colonies [65]. The lung internal structures were segmented by iteratively deploying ant colonies in voxels with intensity above a predefined threshold. Chialvo and Millonas [66] reported that an ant colony moves to a specific destination and releases pheromones based on a set of rules. Iterative thresholding of the pheromone maps was applied to obtain a list of candidates. VBNA uses two different procedures to detect nodules inside the lung parenchyma [67] and nodules attached to the pleura [68]. For the false-positive reduction of LungCAM, they computed a set of 13 features for nodule candidate analysis, including spatial, intensity, and shape features. In order to classify candidates, they used ANN. Their architecture consisted of 13 input neurons, 1 hidden

layer with 25 neurons and 1 neuron in output layer. For the false-positive reduction of VBNA, standard three-layer ANN was used with 12 input nodes, 14 hidden nodes, and 1 output node. They used 12 morphological and textural features extracted from each nodule candidate. Their score was 0.608 in this challenge.

Since lung nodule candidate detection has high commercialization potential, several companies also participated in the challenge in order to test their algorithms. Yet, details of the algorithms were not revealed, they only mentioned that they used CNNs [16].

Some of the groups only participated in the false-positive reduction track.

CUMedVis used the multi-level contextual 3D CNN developed by Dou et al. [69]. They generated three different CNN architectures and used decision fusion (Figure 26).

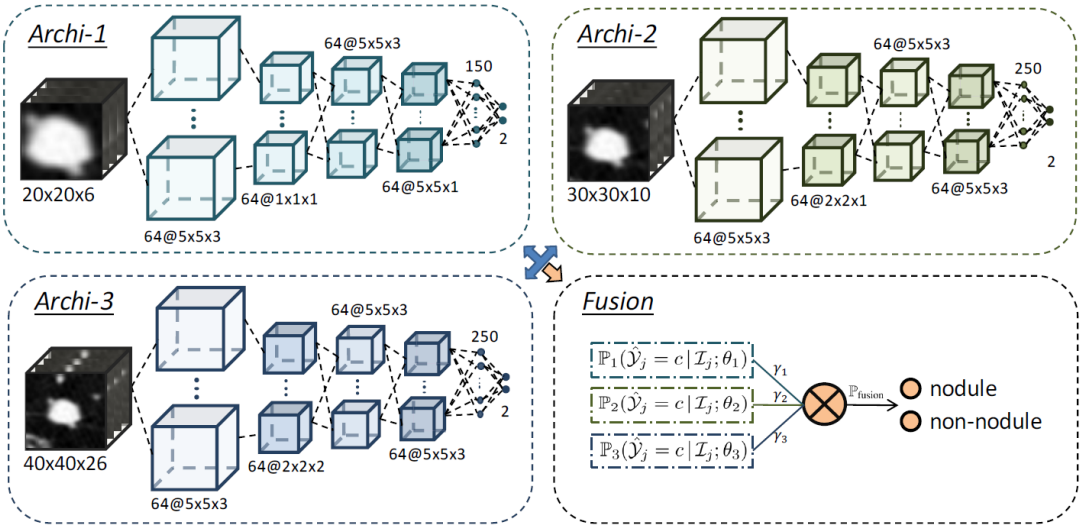

Figure 26: Architectures proposed by Dou et al. [69].

Using three different architectures handles the difficulties that come from the variation of nodule sizes, types, and geometry characteristics. Archi-1 was modified to focus on smaller nodules so that smaller convolutional filters were used in this architecture. Archi-2 and Archi-3 were used to focus on medium and larger nodules respectively. In order to deal with the class imbalance between the false-positives and true-positives, translation (one voxel along each axis) and rotation (90º, 180º, and 270 º) are used for data augmentation. The weights were initialized using Gaussian distribution and were

optimized using the standard backpropagation with momentum. They implemented the system using Theano Machine Learning Library [70] on a GPU NVIDIA TITAN Z. Their overall score was 0.827 [69].

Setio et al. [71] proposed a CAD system based on 2D CNN [71]. For each candidate, they extracted 9 patches of 50x50 mm from different views (Figure 27, a). Their CNN consisted of 3 consecutive convolutional layers and max-pooling layers. The first convolution layer was formed by 24 kernels of 5x5, the second kernel by 32 kernels of 3x3 and the third by 48x48 kernels of 3x3. They used ReLU for activation functions (Figure 10, b). They tested different fusion methods like committee-fusion, late-fusion, and mixed-fusion, which are represented in Figure 27-c. The fusion of the different CNNs was performed using the late fusion method [72] because it generated the best result. They concatenated the fully connected layer of all 9 patches.

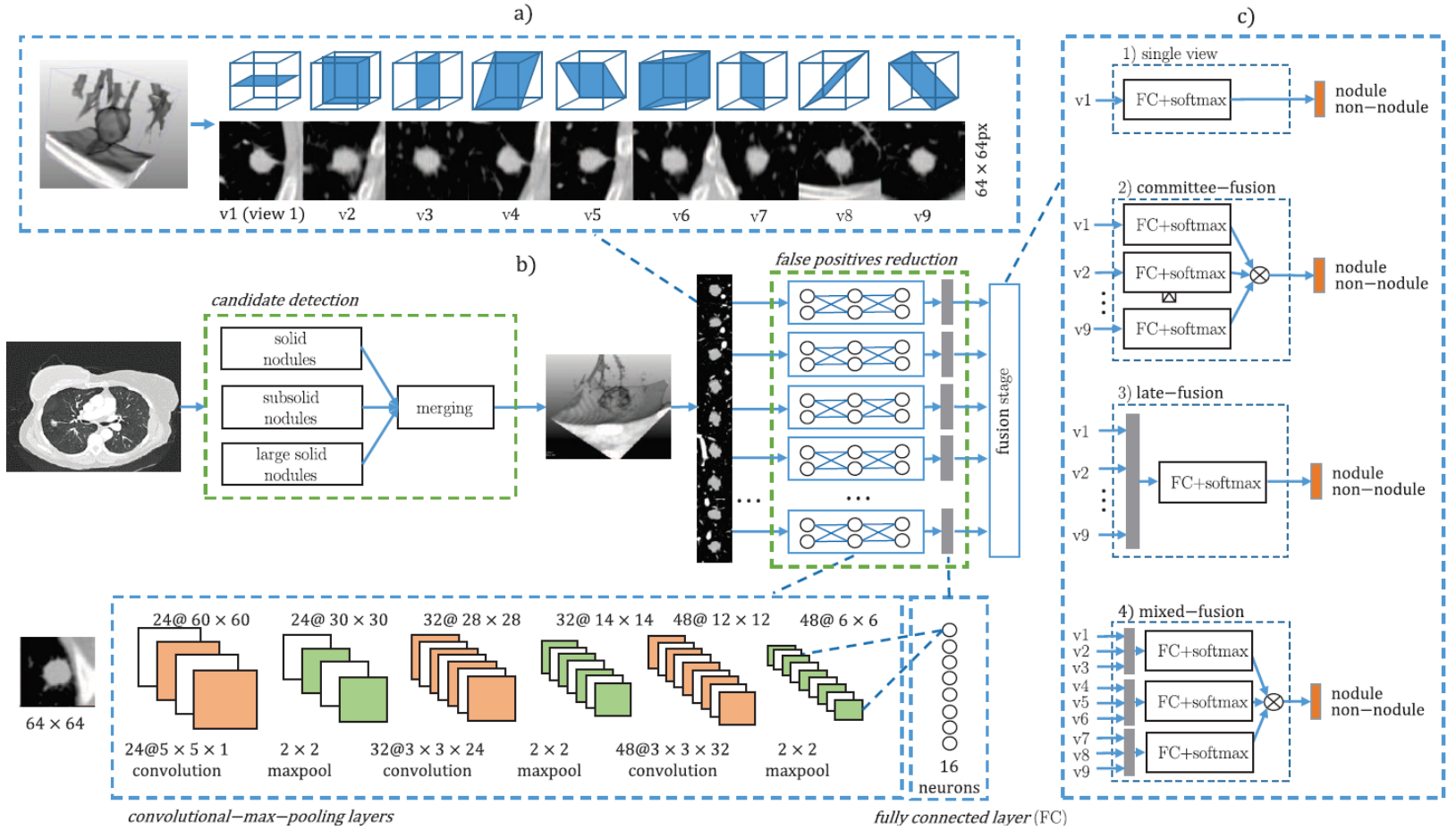

Figure 27: Architecture applied by Setio et al. [71] a) Extracted 2D Patches b) Candidate detection algorithms (Not used in this sub-challenge) c) Fusion methods.

For the data augmentation, they used random zooming [0.9, 1.1] and random rotation [-20º, +20º]. In order to prevent overfitting during the training because of skewed classes, they used random positive and negative candidates with equal distribution. The weights were optimized using RMSProp [47]. Their overall score was 0.828 [71].

CADIMI used multi-slice CNN. For each patch, axial (x and y), sagittal (y and z), and coronal (x and z) views were extracted at three locations: the plane in the exact candidate location, and the planes 2 mm in both directions on remaining free axis (in z, x or y). The network consisted of 2D CNN with three consecutive convolutional layers and max-pooling. The first convolutional layer used 24 kernels of size 5x5; the second used 32 kernels of size 3x3; the third used 48 kernels of size 3x3. The output of the last max-pooling was connected to the fully-connected layer and the fully-connected layer was connected directly to the softmax layer. ReLU was used as the activation function. For data augmentation, vertical/horizontal flips and random cropping to 3x52x52 dimension were applied. Three patches were processed independently and their results were averaged for final prediction value. Weights of the network were initialized using He uniform initialization [73]. For the update rule, they used Nesterov accelerated gradient descent with a learning rate of 0.01, a decay of 10^{-3} and a momentum of 0.9. Their system achieved a score of 0.783 [16].

As in the candidate detection challenge, in addition to the algorithms mentioned above, several individuals and commercial company teams also attended to the challenge. Yet, they did not reveal their algorithm explicitly. It is only known from their brief report that they used 2D and 3D CNN in their algorithms [16].

CHAPTER 3

PROPOSED APPROACH

LUNA16 Challenge dataset is very suitable for deep learning algorithms because dataset consists of many CT images. In addition, there are labeled candidate points, which make the dataset suitable for supervised learning algorithms. That is why all teams that participated in the challenge used CNNs. Yet, as we see in the Background Information chapter, there are many parameters in CNN architecture that require tuning. One of the important parameters is input tensor size. Size of the input changes all other parameters in the model because these parameters must be compatible with each other. In lung CT scans, sizes of the nodule vary from 3 mm to 34 mm and this makes the input size decision even harder. In the literature, there is no objective study that compares the effect of input tensor size to the performance of the system. Therefore, we wanted to observe and compare the effect of input to the performance of the system. In addition, we wanted to show whether different results can be used in decision fusion in order to increase the performance or not. In order to observe these effects, we need to design a finely tuned CNN architecture, which will be suitable for all input sizes.

In this chapter, preprocessing the data and determining the parameters and hyper-parameters of the CNN model are explained in detail. After creating the CNN model architecture, different input sizes are tested on this model and results are presented.

3.1 Dataset

The dataset used in this study is formed by the initial LUNA16 Challenge dataset provided by the organizers and participants' candidate detection algorithms. There are 754,975 candidates and only 1166 out of 1186 nodules are included in these candidates. 20 nodules were not detected by the algorithms; therefore, they are not included in the list. There is a .csv file which is provided with the CT dataset. In this file, location of each candidate and its corresponding class (nodule or non-nodule) is given. Some nodules were detected multiple times in different locations; therefore, there are 1557 true positives in the list. The remaining 753,418 candidate points were labeled as false positive. In other words, only 0.2% of the candidates are true positives, which makes the dataset highly skewed (1:483).

3.2 Preprocessing

Before generating the network architecture, samples in the dataset were examined carefully. Since we planned to use CNN, parameters of the samples such as the distance between the voxels were very important. For the training part, size of the input of the neural network system must be the same for all samples. Yet, while input dimensions are in pixels, real world dimensions are in millimeters. If the same input dimensions correspond to different real-world dimensions, the architecture will not give meaningful results because it will be trained on different volumetric sizes. For example, while a 20x20x20 pixel corresponds to 24x24x16 mm for one scan, it may correspond to 18x18x20 mm for another scan. In order to handle this situation, we examined the voxel spaces in all scans.

In Figure 28, there is a histogram of distances between voxels in X-Y and Z axes. While horizontal axis of the figure corresponds to the distance in mm, vertical axis shows how many samples exist for a certain distance.

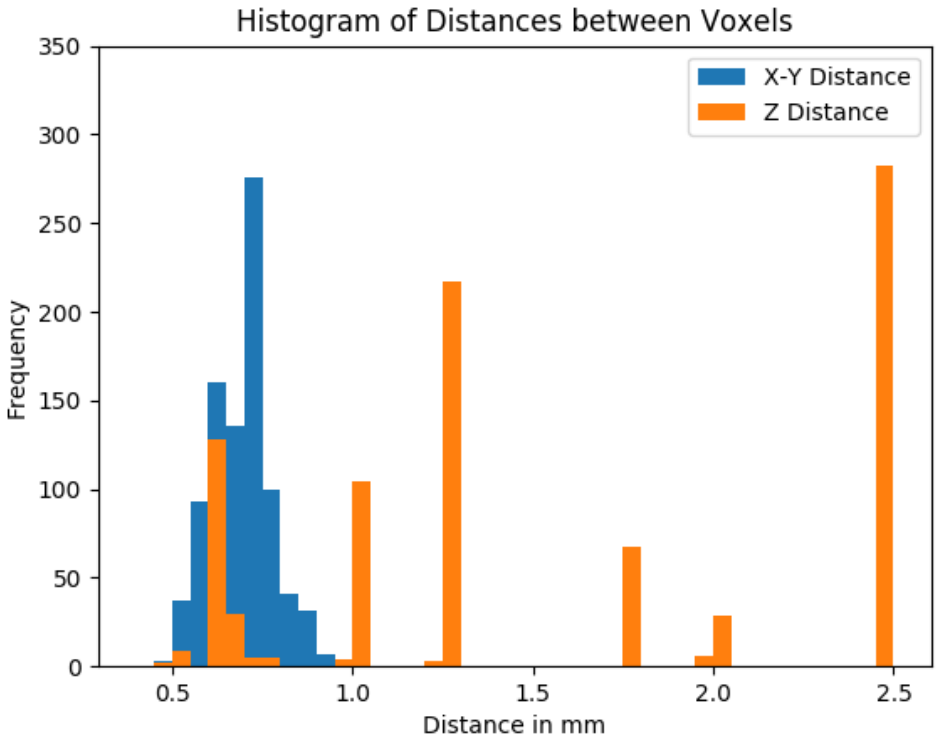

Figure 28: Histogram of distances between voxels in X, Y and Z axes.

It is seen in Figure 28 that voxel distance in the Z axis (between two transverse planes) changes from 0.5 mm to 2.5 mm. On the other hand, voxel distances along the X-Y axes (between two sagittal plane and coronal plane) change from 0.5 mm to 1 mm. When the average distances were measured, it was 0.69 mm for the X-Y dimensions and 1.56 mm for the Z dimension. Although the average distance between two voxels in the Z dimension is 1.59 mm, we set it to 1 mm in order to increase resolution. As a result, all volumes were resampled to new volumes so that voxel distance was 0.7x0.7x1 mm (X, Y, and Z axes, respectively).

This step removed the differences coming from different scans from different scan machines. After this step, the overall sizes of all scans became different. Overall scan size is not important in our algorithm because we focused on the nodules themselves in our network system. The main point is having the same voxel size in different scans so that the same voxel sizes correspond to the same real-world dimensions in all scans. This process was applied by using Python programming language and Numpy library

[74]. Resizing all scans took approximately three days on 8 Intel i7-4790K CPU @ 4.00 GHz processors.

Another important issue with these data was that its output classes were highly skewed to one side (1:483). In a normal training case for this dataset, network parameters would learn the false positive structure because in the training process those cases would be more frequently encountered by the system. In order to prevent this issue, we used data augmentation and modified mini-batch techniques. Since mini-batch is related to training, it is explained in the next section.

For the data augmentation, we took the mirror images with respect to each axis, and rotated the image by 90, 180, and 270 degrees for each case. In total, the number of true positive candidates increased 16 times. In Figure 29 and Figure 30, we show the effects of rotation and taking the mirror of an image, respectively.

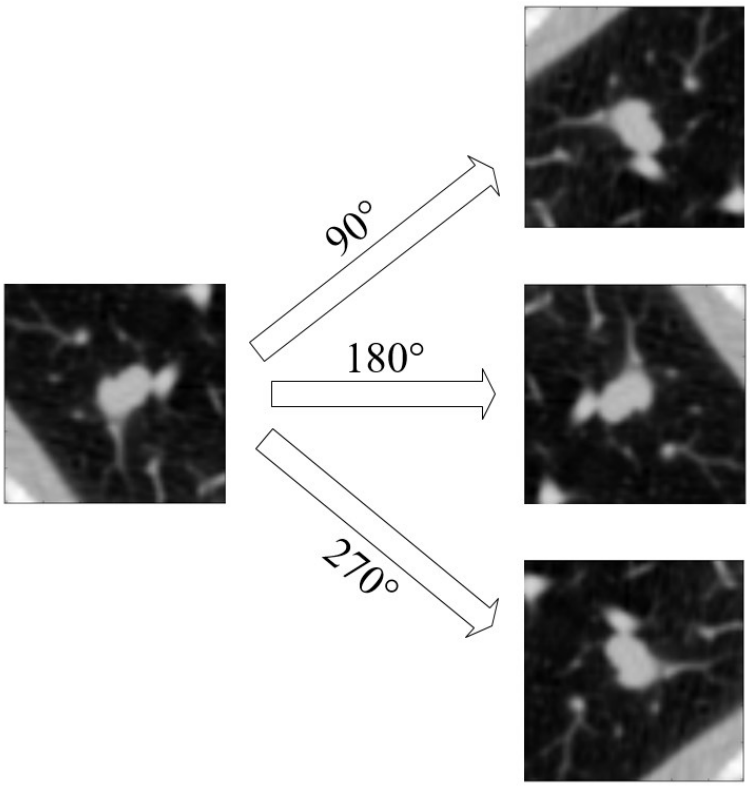

Figure 29: Rotating images in 90, 180, and 270 degrees.

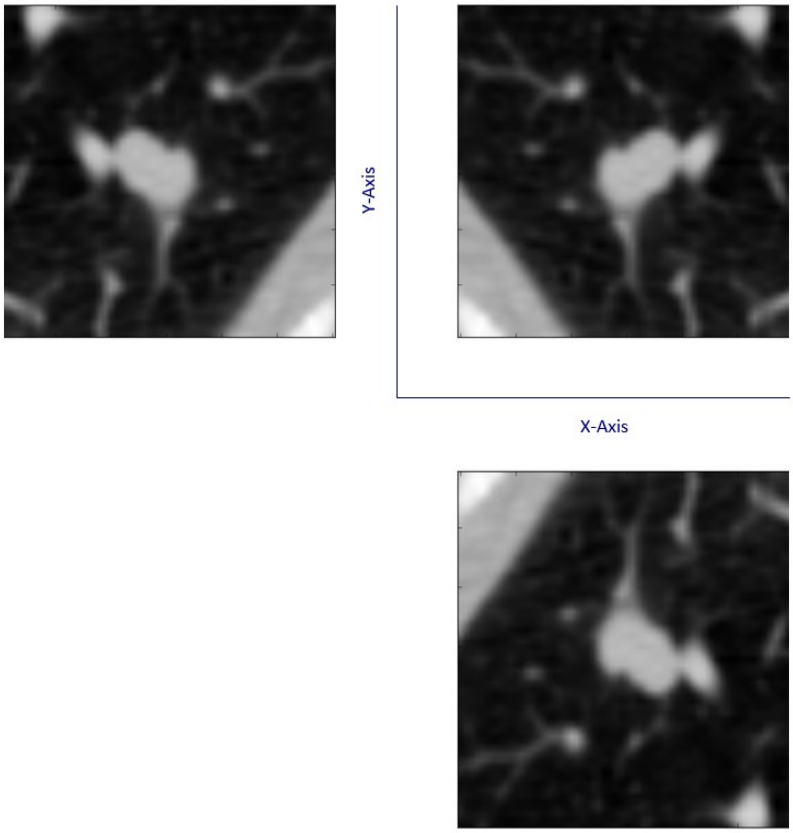

Figure 30: Mirror images with respect to different axes.

The original image and the 3 mirror images make 4 different volumes for a single scan. If each of them is rotated by 90, 180 and 270 angles: 4x4=16 patches can be used instead of a single patch. In total, 1557x16 = 24912 patches are used in the training algorithm. In this case, ratio of the output classes becomes 1:30, but it is still not suitable for training. Therefore, we have solved this issue by using modified mini-batch training algorithm, which will be explained in Section 3.4. Besides, data augmentation was still kept in the system because it improves the generalization ability of the overall model.

3.3 Distribution of the Dataset

In order to focus on both small and large nodules, we used different architectures. With different patch sizes, the model focuses on different characteristics. If the patch size is small, model learns the small nodules better than the large nodules, but only limited contextual information will be used and very large nodules are ignored. If the patch size is too large, noises and redundant particles are also involved in the training patch but patterns of the large nodules will be learned better. In order to determine patch sizes, we examined the histogram of nodule diameters. Dou et al [69], analyzed the size distribution of the pulmonary nodules for all the samples in the dataset (Figure 31).

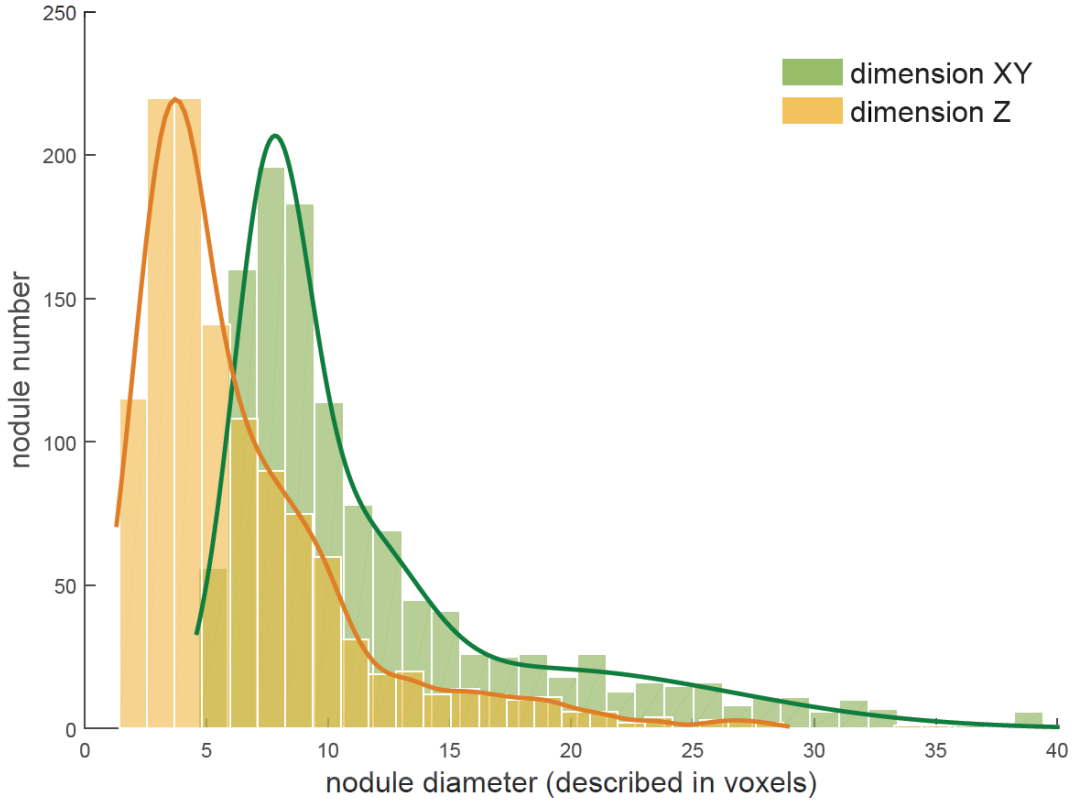

Figure 31: Distribution of sizes of the pulmonary nodules for determining patch sizes [69].

In Figure 31, the diameter of the nodule refers to the volume size. This information is available as part of the challenge dataset. As seen in this figure, volume sizes vary

from 3 mm to 34 mm. Half of the nodules are less than approximately 7 mm. 80% of all nodules are less than 15 mm. We decided to use 5 different patch sizes in order to focus on each target group. We set the smallest patch size as 12x18x18 pixels because this size covers approximately 80% of all nodules. If we had chosen a smaller patch size, it would have discarded many true positive nodules so that the model's sensitivity would be very low. Therefore, this is the smallest size that covers most of the nodules. On the other hand, for the largest patch, we set the size as 36x48x48 voxels. This size is the smallest size which covers all of the nodules in the dataset. Larger sizes than this would be meaningless because this size already covers all of the nodules and increasing the size would bring more redundant data around the nodules. We wanted to determine sizes that are also suitable for our network model, since we have 2 max pooling operations (3x3x3 and 2x2x2) with strides having their dimensions, we kept the size as multiples of 6. As a result, sizes listed in Table 1 were determined.

Table 1: Patch sizes in voxels, and their corresponding real world dimensions in mm.

Patch Size (Voxel)	**Patch Size (Real world size, mm)**
12x24x24	12x16.8x16.8
18x30x30	18x21x21
24x36x36	24x25.2x25.2
30x42x42	30x29.4x29.4
36x48x48	36x33.6x33.6

3.4 Proposed CNN Architecture

Number of convolution layers followed by pooling layers and filter sizes are the two of the most important parameters when tuning the CNN model. Therefore, we have made several experiments with different filter sizes and number of convolutional layers. When conducting these experiments, computational complexities of CNN architectures were designed to be similar. It was possible to create an advanced CNN

architecture but main aim was to tune the model in order to classify input patch sizes. In addition, training a complex CNN model would take too much time for this work.

Using only 1 convolutional layer does not extract enough features. On the other hand, using more than 3 convolution layers increases the complexity and the training time. Therefore, we compared the models, which have 2 and 3 convolutional layers.

As mentioned in the Deep Learning section, max pooling layer, generally, gives better result in most of the cases [75]. Therefore, it is used as a following step to the convolutional layer. Lastly, dropout is used in order to enhance generalization ability of the model. Rest of the model is a standard ANN. There is only one hidden layer after the fully connected layer. One critic step here was to determine number of nodes in the hidden layer. Table 2 to 6 show the architectures of the experimented CNN models. FC Layer in the tables stands for hidden layer in the ANN.

First of all, we created the Model-A (Table 2). For 3 convolutional layers system, filter size of 3x3x3 seemed proper at first. Larger filter sizes could be used but it would increase the complexity of the model. Max pooling size was determined as 2x2x2. When tuning the CNN architecture, patch size of 24x32x32 or 24x36x36 were used because these were the medium patch sizes that we were planning to use in the training.

Table 2: CNN architecture of the Model-A.

Model-A
Patch Size = 24x32x32
C1 32@3x3x3
Max Pooling (2, 2, 2)
Dropout 0.2
C2 32@3x3x3
Max Pooling (2, 2, 2)
Dropout 0.2
C3 32@3x3x3
Max Pooling (2, 2, 2)
Dropout 0.2
FC Layer: 200

Model-A gave a score of 0.681. This score seemed promising initially; therefore, we modified the other models upon this structure. In order to compare this model with another model that has 2 convolutional layers, we created the Model-B (Table 3). We increased the number of convolutional filters and filter size in order to make the complexities of the two model same. We could use 32 filters in the convolution operations but that would make the Model-B minimized version of Model-A. In Model-B, 64 filters which were of size 3x5x5 were used. In addition, first max pooling was changed to 3x3x3 because if we had used a filter size of 2x2x2, there were going to be too many nodes at the first fully connected layer (input layer of ANN).

Table 3: CNN architecture of Model-B.

Model-B
Patch Size = 24x36x36
C1 64@3x5x5
Max Pooling (3, 3, 3)
Dropout 0.2
C2 64@3x5x5
Max Pooling (2, 2, 2)
Dropout 0.2
FC Layer: 400

Model-B gave a score of 0.705. More number of convolutional filters and bigger filter sizes improved the performance. This shows that there are complex patterns in the nodule that can be captured with bigger filters. When we saw an increase in the performance, we wanted to observe whether 2 hidden layers instead of a single hidden layer would generate better results; therefore, we used two hidden layers instead of one. Their number of nodes was 600 and 250, respectively (Model-C, Table 4).

Table 4: CNN architecture of Model-C.

Model-C
Patch Size = 24x36x36
C1 64@3x5x5
Max Pooling (3, 3, 3)
Dropout 0.2
C2 64@3x5x5
Max Pooling (2, 2, 2)
Dropout 0.2
FC Layer: 600
FC Layer: 250

We got a score of 0.546 from Model-C. Performance drastically decreased when it was compared with the Model-B. It was observed that 2 hidden layers create an overfitting situation; therefore, increasing the fully connected size was not a good idea for this architecture. After having this result, we wanted to observe if there is an increase in performance when we decrease the number of nodes in the hidden layer of the Model-B. Therefore, we created another model that we have named Model-D, which has 200 nodes in its hidden layer. (Table 5).

Table 5: CNN architecture of Model-D

Model-D
Patch Size = 24x36x36
C1 64@3x5x5
Max Pooling (3, 3, 3)
Dropout 0.2
C2 64@3x5x5
Max Pooling (2, 2, 2)
Dropout 0.2
FC Layer: 200

Model-D gave a score of 0.736. We realized that our initial number of nodes in the hidden layer of Model-B was high.

We also wanted to observe what would be our performance if we had used more convolutional filters with a filter size of 3x5x5 in the first convolution layer of Model-A. In order to observe this change, we created the Model-E (Table 6).

Table 6: CNN Architecture of Model-E.

Model-E
Patch Size = 24x32x32
C1 48@3x5x5
Max Pooling (2, 2, 2)
Dropout 0.2
C2 32@3x3x3
Max Pooling (2, 2, 2)
Dropout 0.2
C3 32@3x3x3
Max Pooling (2, 2, 2)
Dropout 0.2
FC Layer: 200

Model-E gave a score of 0.689. Which is very close to the Model-A. We concluded that increasing the filter size and number of filters in the first convolutional layer does not provide an increase in performance. Figure 32 to 36 show FROC curves of the experiments conducted above.

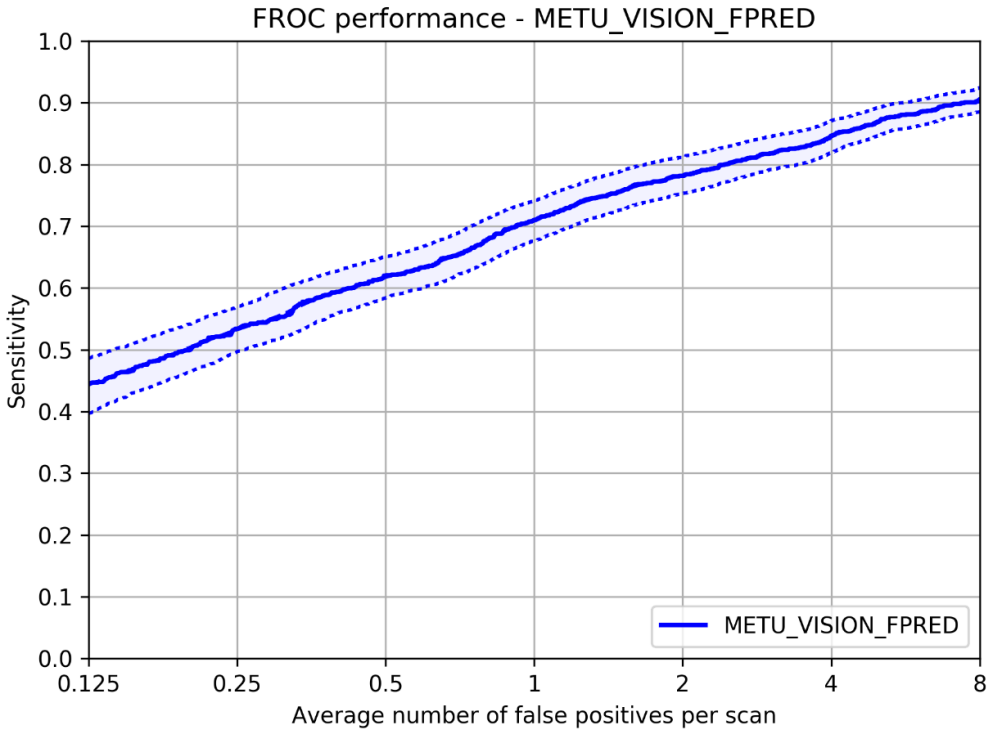

Figure 32: FROC performance of Model-A.

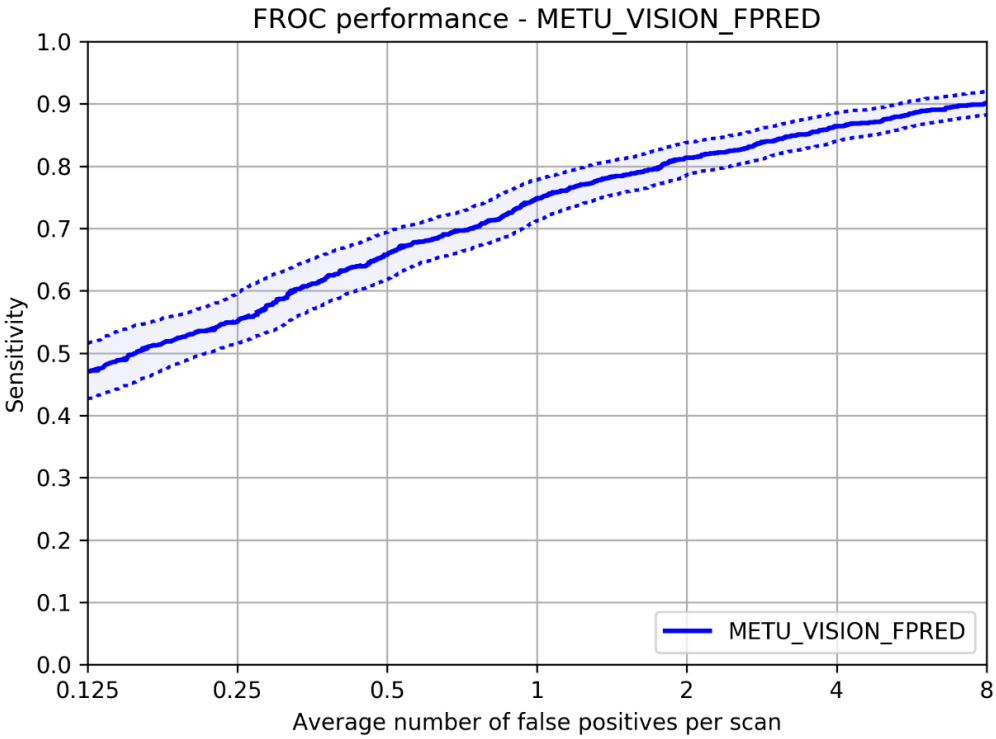

Figure 33: FROC performance of Model-B.

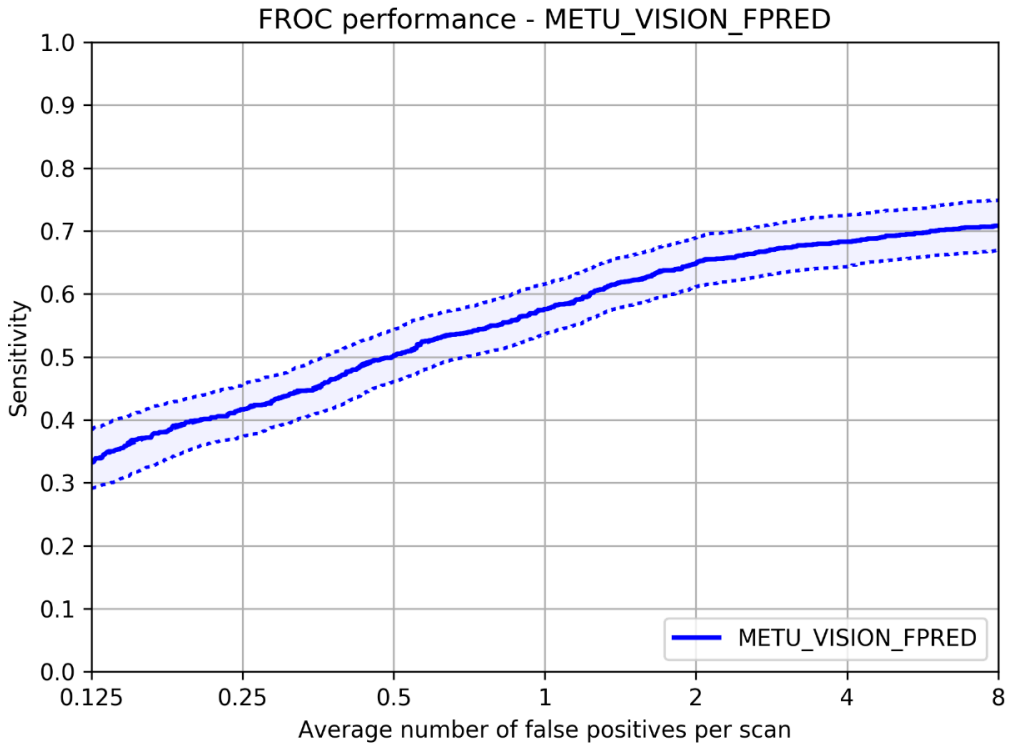

Figure 34: FROC performance of Model-C.

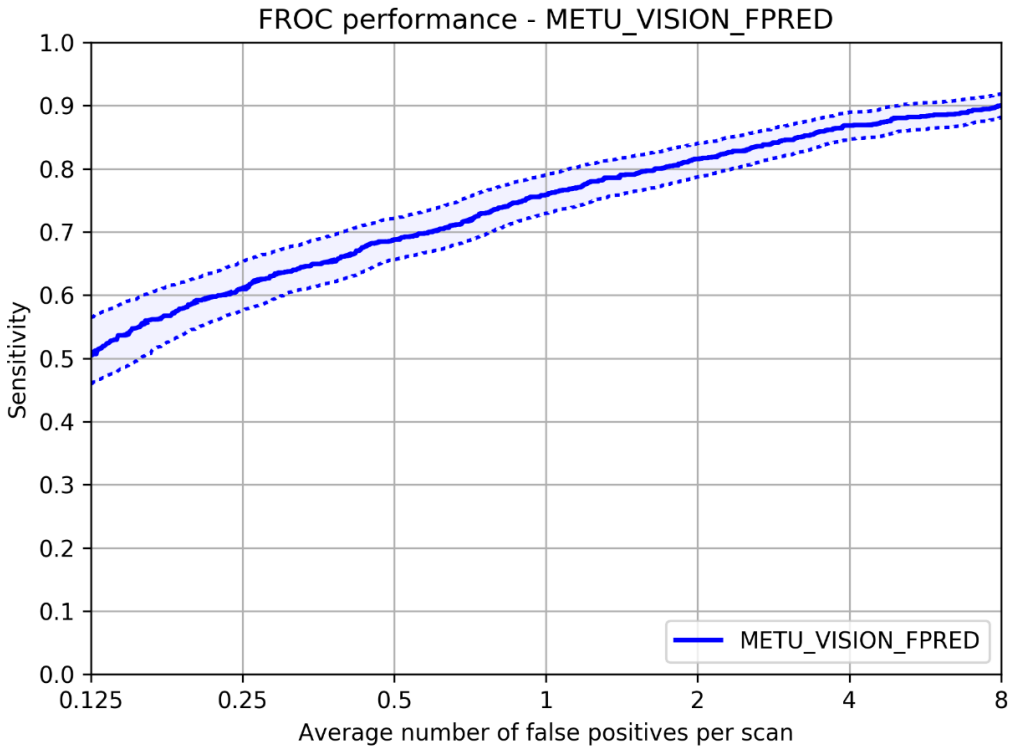

Figure 35: FROC performance of Model-D.

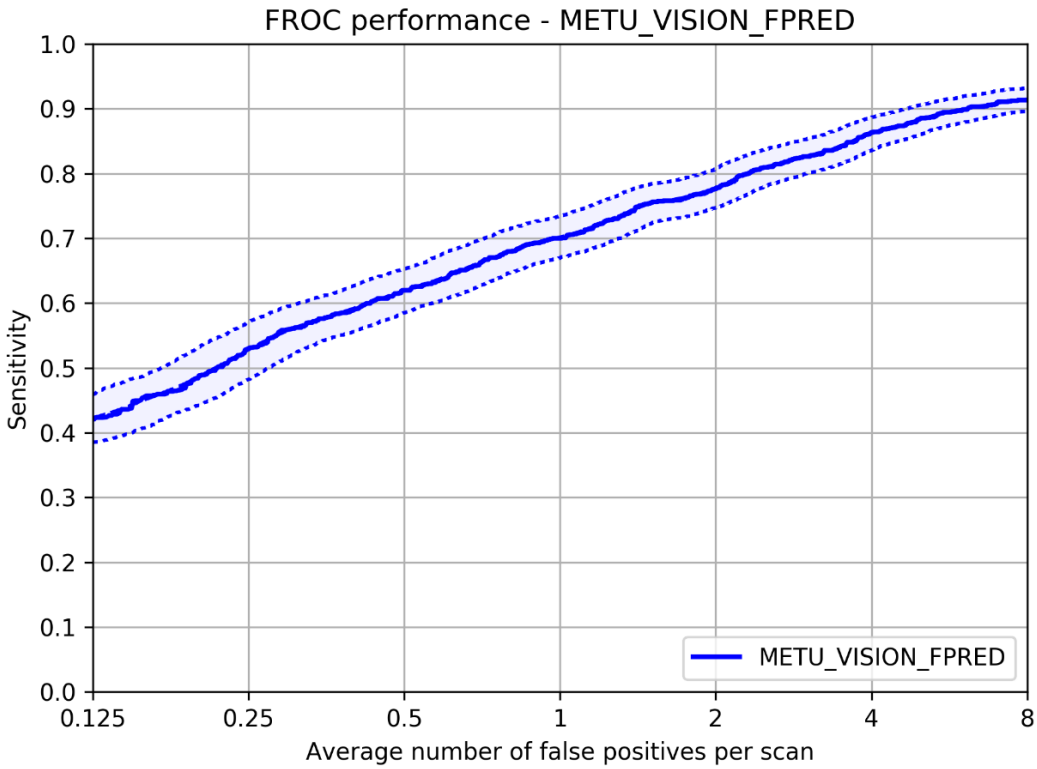

Figure 36: FROC performance of Model-E.

2 convolutional layers with 3x5x5 filter size, which is model D, gave the best result. In light of these experiments, models in Table 7 were used in the training step. Since input patch sizes are different from each other, number of nodes in the hidden layer (first fully connected layer) was adjusted according to their input patch sizes. In Table 7, last fully connected layer stands for the output layer.

Overall structure can be represented as follow:

$$Data \rightarrow [C \rightarrow ReLU \rightarrow MaxPool \rightarrow Dropout]^2 \rightarrow FC$$

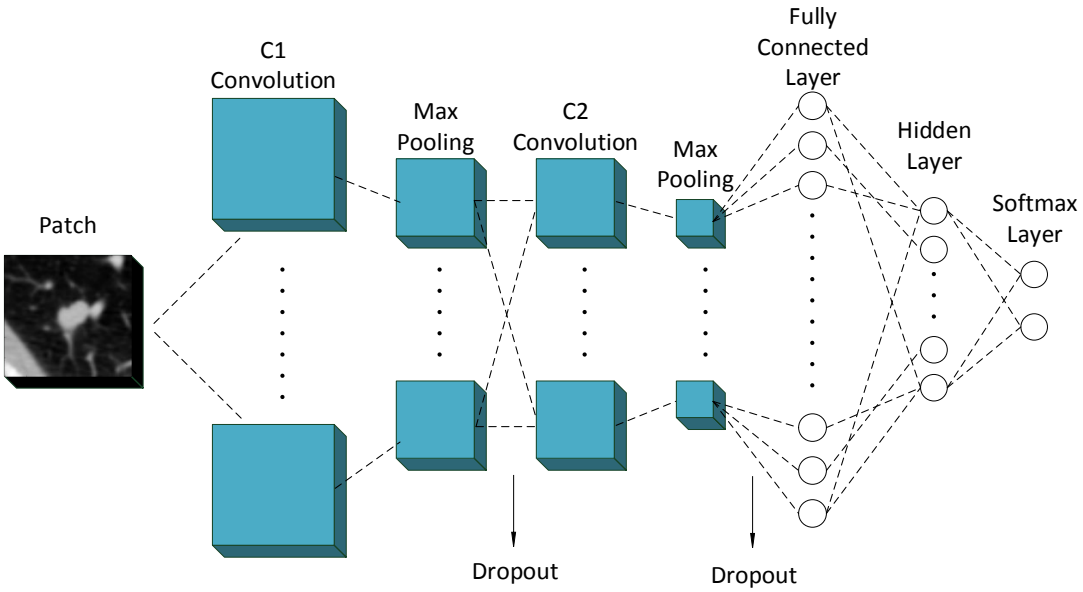

Figure 37: Generic model architecture.

Table 7: Architectures of the CNN models compared in this study.

Model-1 Patch Size: 12x24x24	**Model-2** Patch Size: 18x30x30	**Model-3** Patch Size: 24x36x36	**Model-4** Patch Size: 30x42x42	**Model-5** Patch Size: 36x48x48
C1 64@3x5x5	C1 64@3x5x5	C1 64@3x5x5	C1 64@3x5x5	C1 64@3x5x5
Max Pooling (3, 3, 3)	Max Pooling (3, 3, 3)	Max Pooling (3, 3, 3)	Max Pooling (3, 3, 3)	Max Pooling (3, 3, 3)
Dropout 0.2	Dropout 0.2	Dropout 0.2	Dropout 0.2	Dropout 0.2
C2 64@3x5x5	C2 64@3x5x5	C2 64@3x5x5	C2 64@3x5x5	C2 64@3x5x5
Max Pooling (2, 2, 2)	Max Pooling (2, 2, 2)	Max Pooling (2, 2, 2)	Max Pooling (2, 2, 2)	Max Pooling (2, 2, 2)
Dropout 0.2	Dropout 0.2	Dropout 0.2	Dropout 0.2	Dropout 0.2
Fully Connected Layer: 100 nodes	Fully Connected Layer: 150 nodes	Fully Connected Layer: 200 nodes	Fully Connected Layer: 350 nodes	Fully Connected Layer: 500 nodes
Fully Connected Layer: 2 nodes	Fully Connected Layer: 2 nodes	Fully Connected Layer: 2 nodes	Fully Connected Layer: 2 nodes	Fully Connected Layer: 2 nodes
SoftMax	SoftMax	SoftMax	SoftMax	SoftMax

Main aim was to compare the effect of input patch sizes to the performance; therefore, main structure of the CNN architecture is the same for all patch sizes (Figure 37). We deliberately wanted to keep the model structure similar, because if they were different from each other, effect of the volume size in training would be impossible to compare. Since patch sizes are different for each model, the number of nodes in the first fully connected layer is different in each network. The number of nodes in the first fully connected layer directly affects the number of nodes in the hidden layer. Too many nodes in the hidden layer will make the network lose its generalization ability. On the other hand, with too few nodes, the network has to use too little information and may not solve the complex issues [76]. Therefore, the same number of nodes in the hidden layer cannot be used for all architectures. The only difference between models is the number of nodes in the hidden layer (first fully connected layer).

ReLU [44] was used as the activation function. For optimization method, Adam [48] adaptive learning method was chosen. For optimization, cross-entropy loss function was used in order to estimate the error in the last layer:

$$E = -\sum_{i} y_i' \log(y_i), \tag{22}$$

where y is the predicted probability, and y' is the true value of the class of the sample. Cross-entropy loss function mainly punishes larger errors more. If the predicted probability and the true output are very close to each other, error will be close to zero. On the other hand, if the difference between the predicted value and the true output gets larger, the error will be increased exponentially. This system accelerates the learning mechanism.

When training and testing the algorithm, 10 fold cross-validation was used. Although data augmentation was used in order to prevent class imbalance, ratio of the false positives to true positives was still skewed (1:30), as explained in Section 3.2. In order to handle this class imbalance issue, mini-batch was used. Following is the modified mini-batch algorithm; flowchart of the algorithm is also given in Figure 38:

1. Collect all true positives from the training set.
2. Collect the first N false positives where N is the number of all true positives for that training set.
3. Mix the set of true positives and false positives, train the mixed set.
4. When training finishes, get the next N false positives and repeat step 3. Repeat this step until the training process converges.

Since there are 754,975 candidates in total, for each of the 10 subsets, there are approximately 75,000 candidates. When one of these subsets is used as the test set, we have around 675,000 candidates that remain for the training set. For each training set, we have around 1400 true positives (Since there are 1557 true positives in the dataset, approximately 90% of them remains for the training set). That means, we need to include the first 1400 false positives in the candidate list, and merge these two sets to form our first mini-batch. When the training finishes for that mini-batch, we collect the next 1400 false positives, and apply the training again. During the experiments, we realized that after including the first 150,000 candidates, there is no more advancement in the training accuracy. Therefore we terminated the training process when the number of candidates exceeded 150,000 in order to prevent the network from memorizing certain patterns. This number is approximately 22% of the overall training data. When training the mini-batch, we cannot train all set as a whole due to memory constraints. Therefore, we trained the mini-batch by batches that have 32 samples.

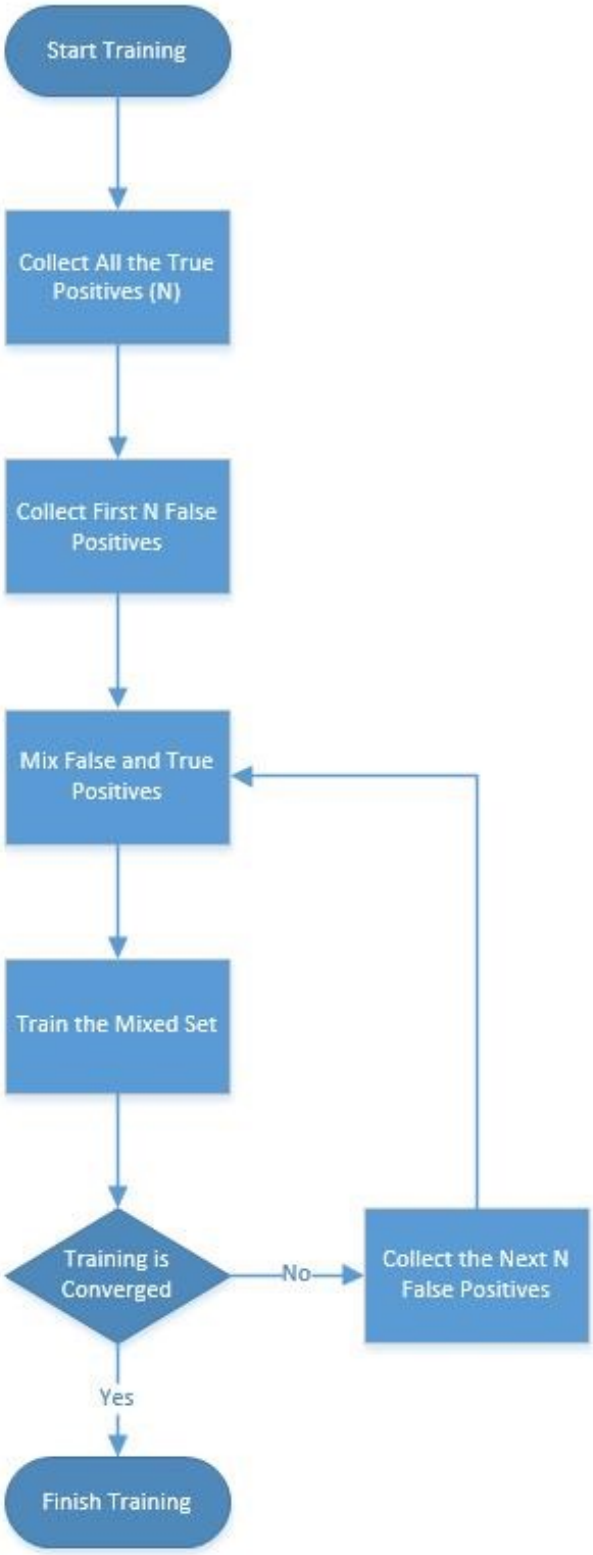

Figure 38: Modified Mini-batch algorithm flowchart.

The networks were implemented in Python programming language using deep learning library Keras [77]. The algorithm was run on NVIDIA TITAN X GPU. Patch size highly affected the training time. As the patch size was increased, more time was spent on the training process. On average, training for a single patch took approximately 30 hours.

3.5 Ensemble of Models for Decision Fusion

Ensemble of classifiers is a powerful technique in machine learning models. Different models can focus on different characteristics in the input data and their strengths can be used together. In this thesis, we wanted to use this technique in order to observe whether there is an increase in the overall score. We experimented with decision fusion of different combinations of models in order to get a benchmark. In this fusion step, output probabilities of the models were simply averaged. Experimented fusions are specified in Table 8.

Table 8: Models included in the fusions.

Fusions	Model-1	Model-2	Model-3	Model-4	Model-5
Fusion-1	✚		✚		✚
Fusion-2	✚	✚			✚
Fusion-3	✚			✚	✚
Fusion-4	✚	✚	✚	✚	✚

Results of these fusions are presented in the next chapter.

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, first of all, we evaluated the individual performance of each model introduced in Table 7. What we were aiming was to measure and compare the effect of different patch sizes. Our hypothesis was that performance of the models would increase with the increasing patch size because larger patch sizes cover more nodules compared to smaller nodules.

Model results were evaluated with a framework supplied by the challenge organizers. The framework was written in Python programming language. The output of the framework is FROC curve as explained in Section 2.4. As an output, maximum, mean, and minimum sensitivities of the false positives per scan starting from the 0.125 to 8 were provided by the framework.

Figure 39 to 43 show FROC curves of the 5 different models.

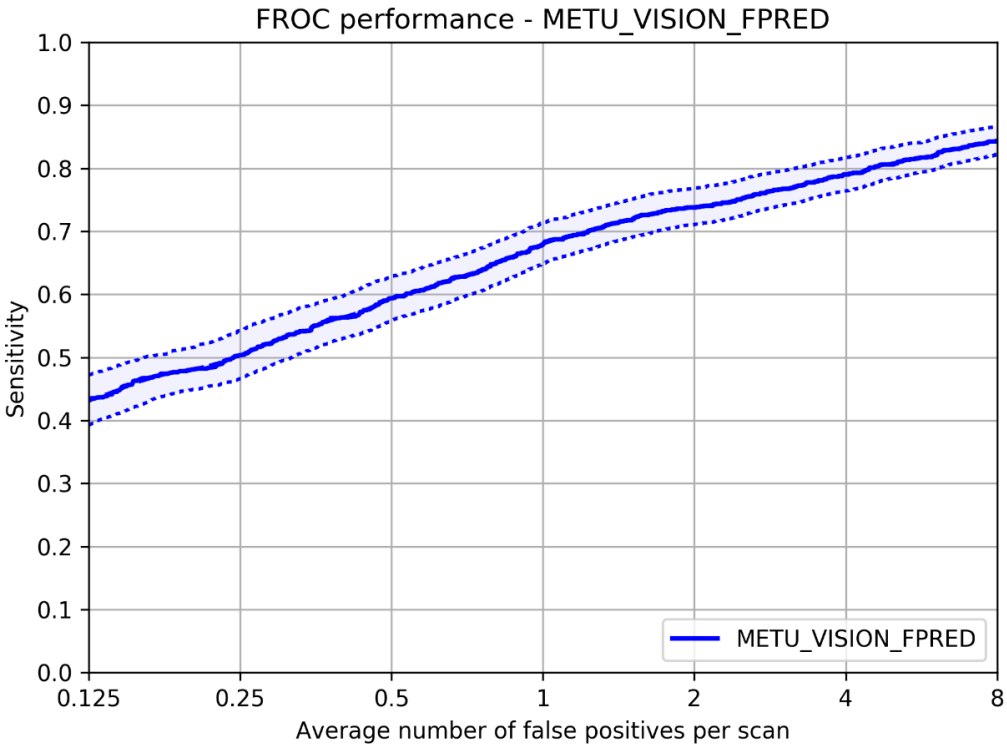

Figure 39: FROC performance of Model-1.

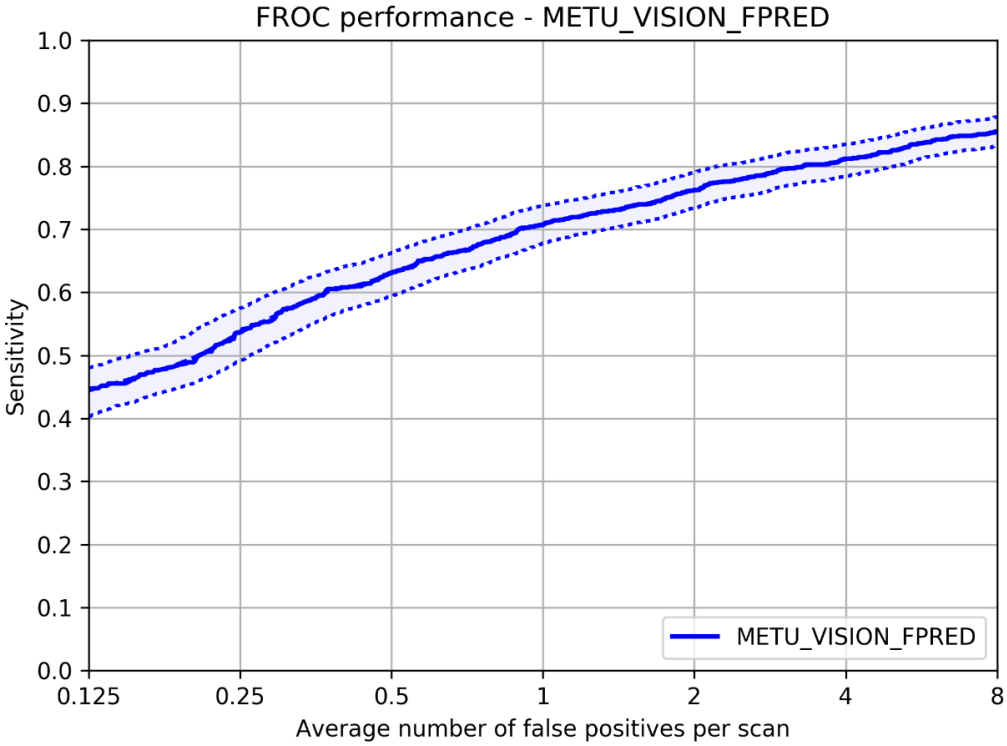

Figure 40: FROC performance of Model-2.

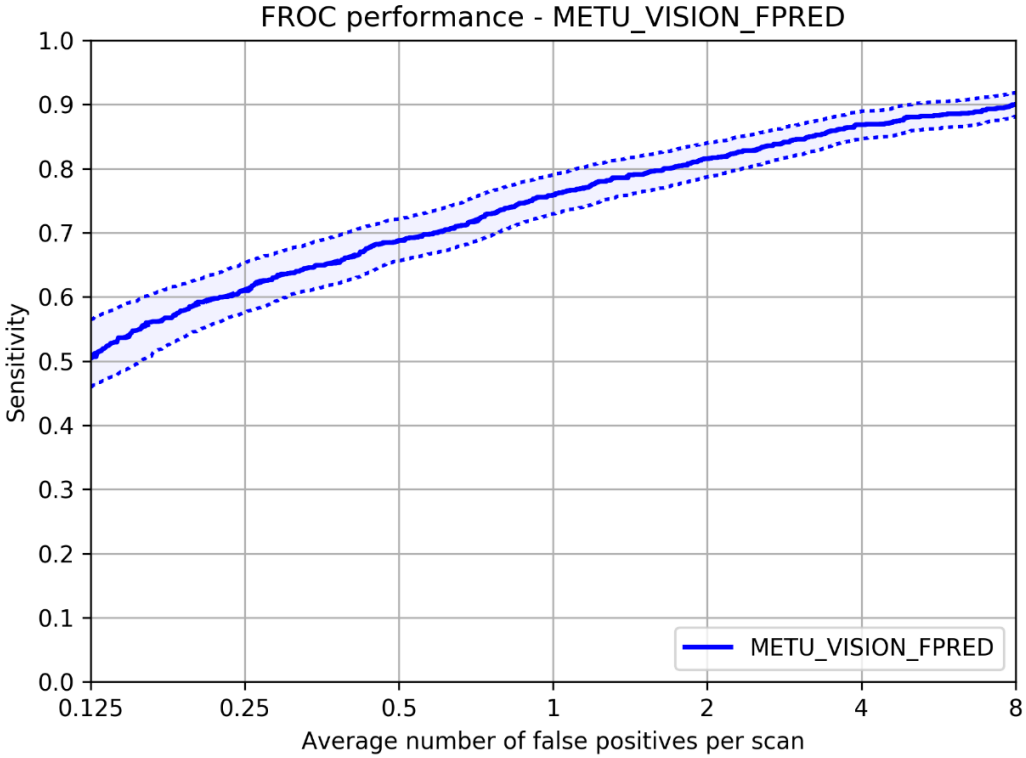

Figure 41: FROC performance of Model-3.

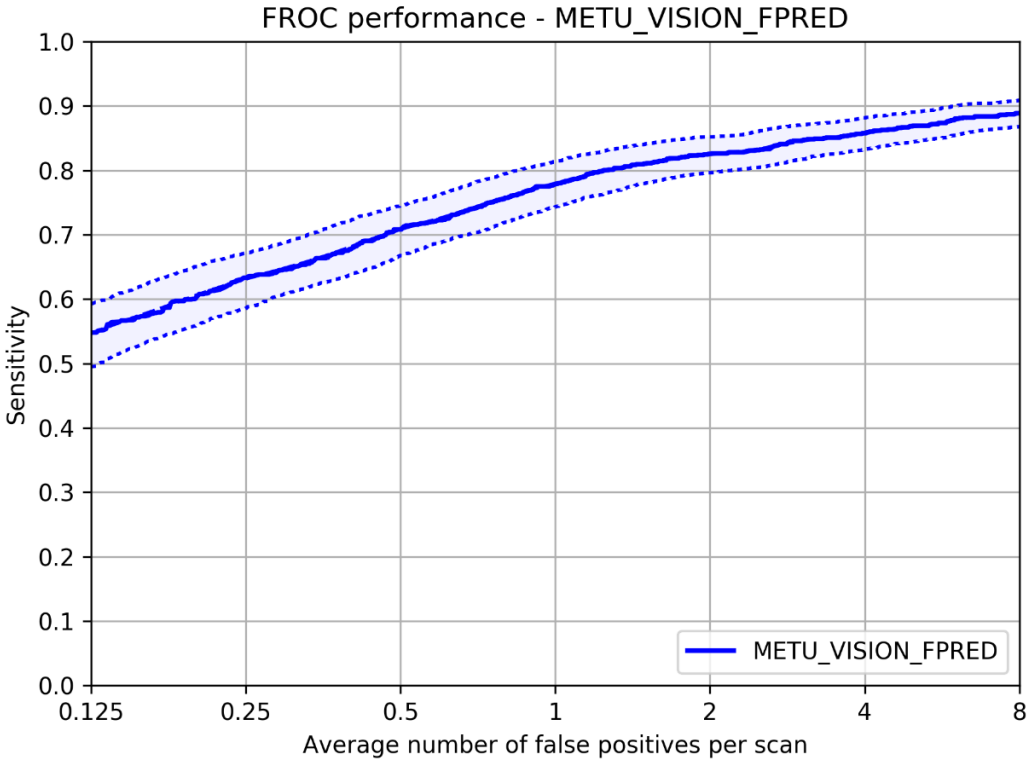

Figure 42: FROC performance of Model-4.

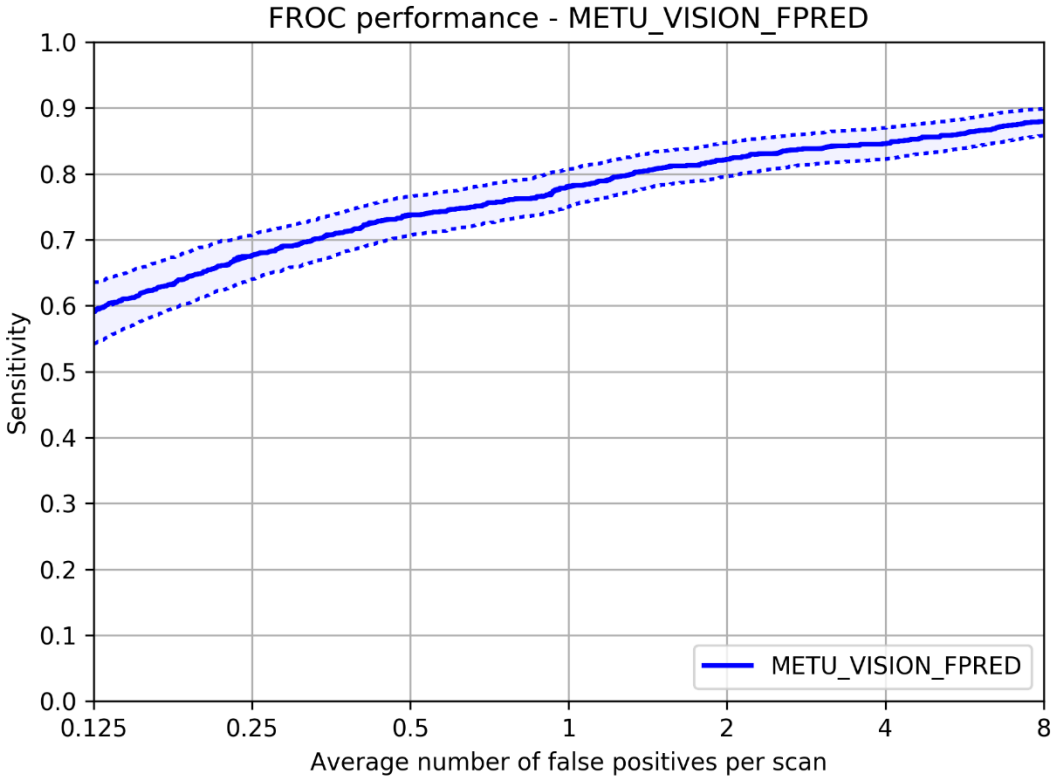

Figure 43: FROC performance of Model-5.

It is seen in the figures that as the patch size increases, the overall score also increases because larger patch sizes cover more nodules in the dataset. When the average number of false positives per scan is 0.125, sensitivities are between 0.45 and 0.6. Although false positive rate is very low, sensitivities are also very low, which is not applicable to clinical use. On the other side, when the average number of false positive per scan is 8, sensitivities are between 0.85 and 0.9. This means that approximately 90% of all the true positives can be detected in the dataset successfully. Yet, there are also false positives in the detected nodules and on average there are 8 false positives per scan. Dataset consists of 888 scans so that 7104 (888x8=7104) false positives are generated in the classification when the sensitivity is between 0.85 and 0.9.

Mean sensitivity values at seven false positive rates and scores for each model are shown in Table 9.

Table 9: Sensitivities of the models at seven false positive rates and model scores.

Model No	0.125	0.250	0.5	1	2	4	8	Score
1	0.431	0.504	0.593	0.68	0.739	0.79	0.844	0.654
2	0.444	0.537	0.631	0.709	0.763	0.811	0.856	0.679
3	0.51	0.613	0.689	0.76	0.815	0.868	0.9	0.736
4	0.547	0.632	0.708	0.78	0.825	0.858	0.89	0.749
5	0.591	0.676	0.736	0.779	0.822	0.847	0.879	0.761

Table 9 shows the scores corresponding to 5 different models introduced in Section 3.4. The score was calculated as mentioned in the Section 2.4. Each patch generated different results for similar CNN architectures. When Model-1 and Model-5 are compared, there is a 0.107 point difference in their scores, which is a huge gap for very similar models. This situation shows that trained volume size is very important for these kinds of nodule detection problems. Even a single model that produces good results must be compared with the same model that uses different patch sizes.

It is seen in Table 9 that with the increasing patch size, the performance of the model also increases. Although smaller patch sizes have lower accuracy, they have a better candidate detection probability for smaller nodules. In their receptive field, there is only room for one nodule. Table 10 shows probabilities of four randomly selected small nodules that have sizes around 5 mm. It can be seen that when nodules are small, smaller patch size generate better result compared to larger patch size. For large patch sizes, small nodules with many noise and residues around will be hard to distinguish. Meanwhile, large nodules cannot be covered by the smaller patches; in order to catch them, larger volumes must be used.

Table 10: Probabilities of the smallest and the largest patches for being a nodule for four sample nodules.

Nodule Diameter (mm)	Model-1 Probability	Model-5 Probability
4.54	0.998	0.722
4.42	0.359	0.144
5.1	0.997	0.846
5.23	0.957	0.0001

Our second hypothesis was to increase the overall performance by using the ensemble of classifiers. Table 11 gives the sensitivity values of the seven predefined false positive rates and averages of these sensitivities for four different fusion scenarios.

Table 11: Sensitivities of the fusions at seven false positive rates and fusion scores.

Fusion No	0.125	0.250	0.5	1	2	4	8	Score
1	0.57	0.654	0.737	0.802	0.855	0.89	0.915	0.775
2	0.548	0.633	0.722	0.784	0.840	0.873	0.892	0.756
3	0.577	0.681	0.761	0.815	0.860	0.887	0.904	0.784
4	0.588	0.669	0.749	0.831	0.863	0.892	0.913	0.786

As it is seen in Table 11, fusions, except for Fusion-2, increased the overall sensitivities when compared to any single model. Since Fusion-2 mainly consists of models which use smaller patch sizes, there is no performance improvement. Yet, it is still very close to the best model, which is Model-5. The best result, which is Fusion-4, is achieved by combining all models. The Fusion-4 increased the score of Model-5 by 0.025 points.

Figure 44 gives the FROC plot for 5 model and their fusion. This plot shows that the fusion of the all models generates the best performance. At first, this situation may seem counterintuitive, because one can assume that when taking the average of the

probabilities of all models, the result should be the average of their individual results. Yet, in reality, the average of probabilities performs better than the best model.

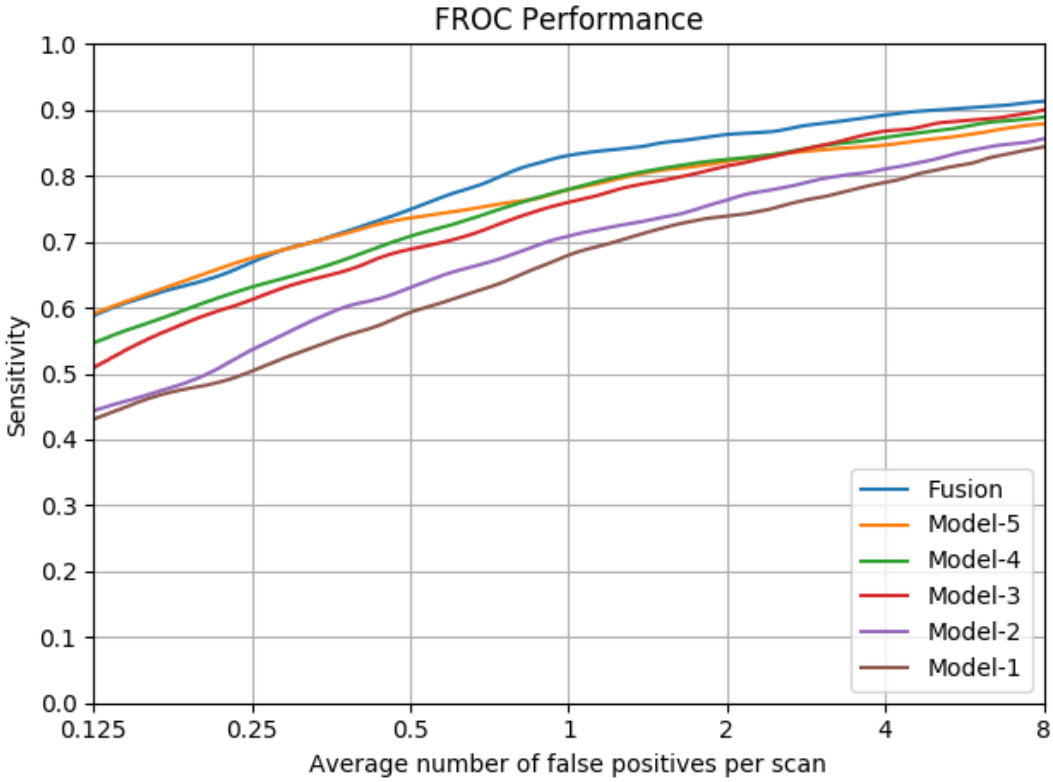

Figure 44: FROC performance of Fusion-4

Decision fusion combines the strength of different models. Scores of Fusion-1 and Fusion-3 are very close to the Fusion-4. We can say that any reasonable fusion boosts the overall performance, but the fusion that includes all models is superior to others because it overcomes weaknesses of the models better than other fusions, which use less number of models. FROC performances of the fusions are given in Figure 45 to 47.

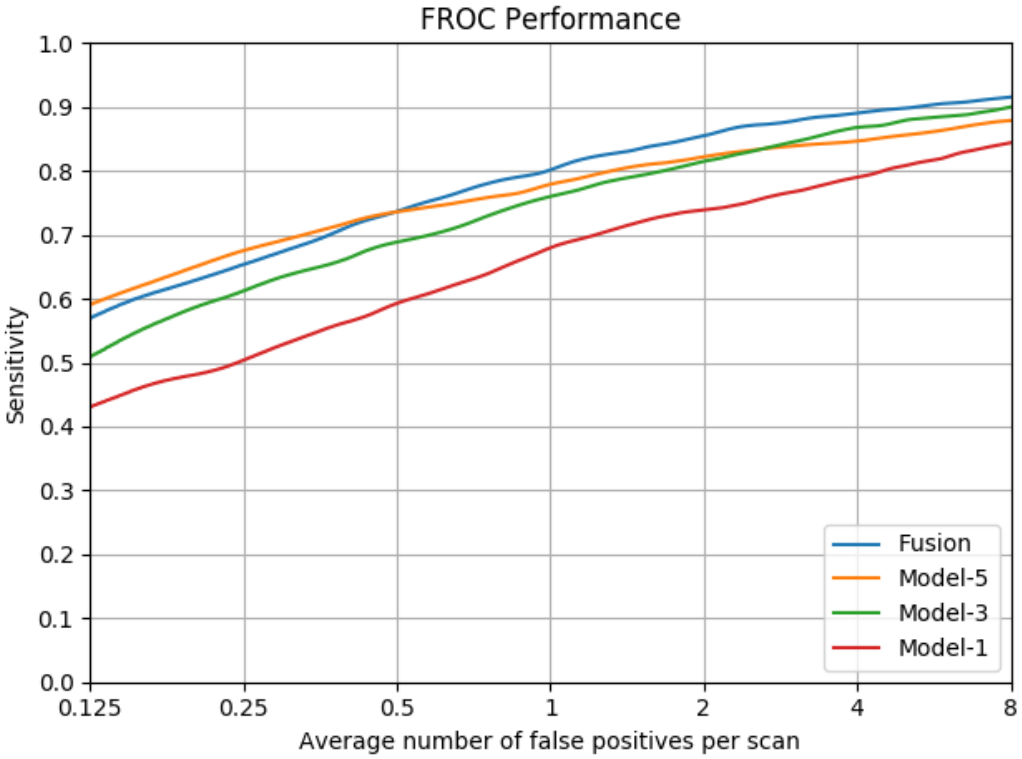

Figure 45: FROC performance of Fusion-1.

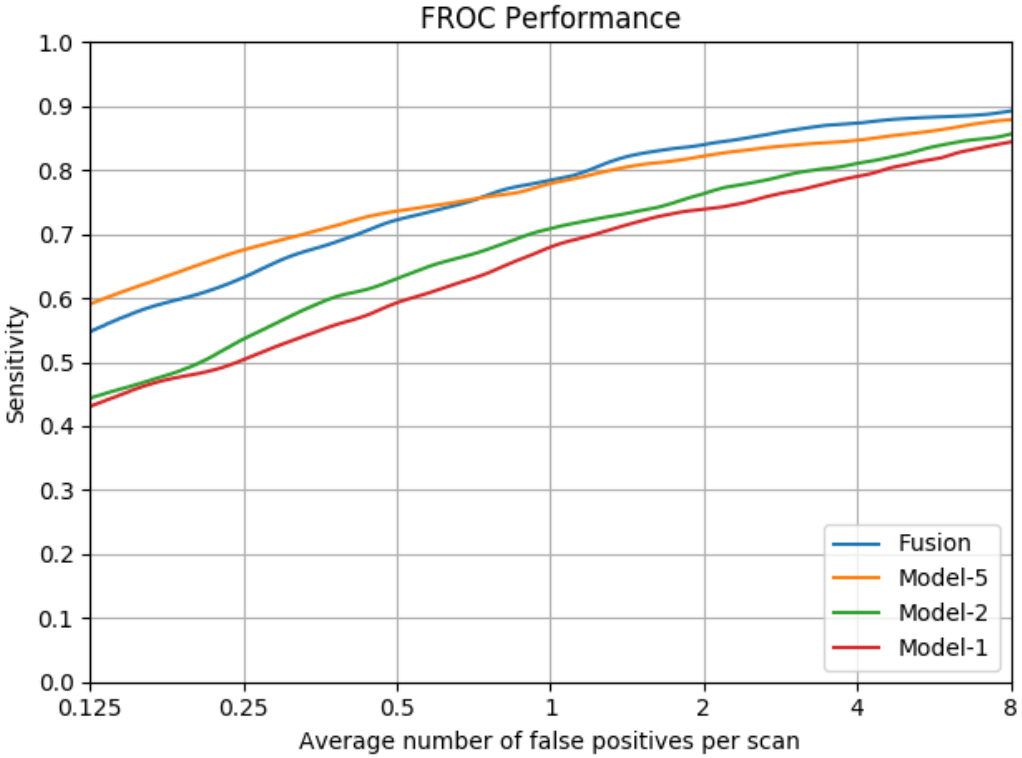

Figure 46: FROC performance of Fusion-2.

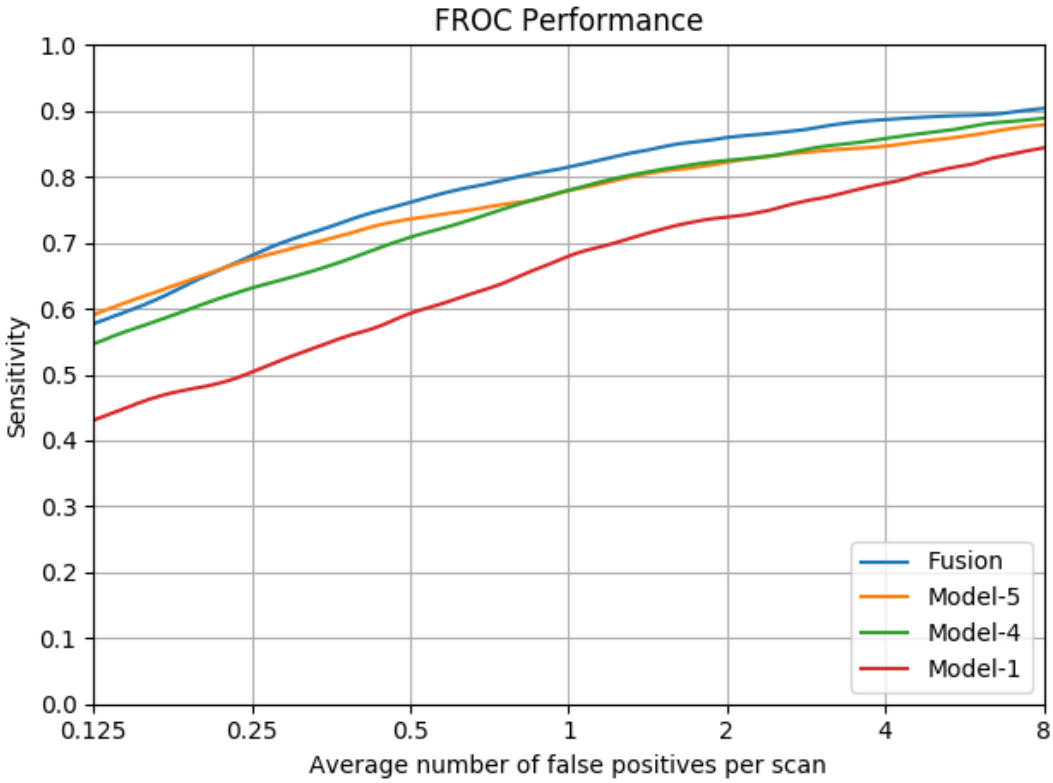

Figure 47: FROC performance of Fusion-3.

Table 12 shows some of the selected research that is focused on the false positive reduction for LUNA16 Challenge. Although there are better submissions on the score list, they have not been included because their methods are either not clear or missing; therefore, it is impossible to compare. LUNA16 Challenge was an open challenge that was different from many challenges in medical image analysis. The data used in the challenge and annotations are publicly available. Therefore, there is not a hidden test set to evaluate algorithms objectively. Recently, there were some submissions that have a very high accuracy but do not have a detailed explanation about the algorithm. Challenge organizers stated that some of the submissions are 'too good to be true' so that they closed the submissions without a paper that fully explains the overall algorithm. Moreover, they stated that they have some sanity checks to evaluate results and some submissions were found to be tuned for the data in order to get good result. For example, some locations included in the candidate list were labeled as non-nodule in the ground truth, but, in fact, they were nodule. If a submission with very good

performance gives these locations an extremely low score for being a nodule, they think that it is fishy. Therefore, only the submissions that have a detailed research paper are included in the Table 12.

Table 12: Methods and their scores.

Method	Score
PATech [16]	0.968
Dou et al. [69]	0.827
Setio et al. [71]	0.814
Our Proposed Method	0.786
CADIMI [16]	0.784
ZNet [16]	0.758
Dobrenkii et al. [78]	0.735
LungNess [16]	0.637

A recent submission, which was submitted on 20 December 2017, from PATech (Ping An Technology [79]) outperformed all other submissions [16]. They used a 3D CNN architecture consisting of 27 layers. 6 of the layers were convolutional layers and they used a different loss function called focal loss, which is good for unbalanced classes. They used 4 high performance Tesla K80 GPUs in order to train the algorithm. This situation shows that hardware is very important in deep learning training. Even if there is time for training, this architecture cannot be trained on low level GPUs because of the memory consumption during training.

Second best method uses 3 convolutional layers [69]. In addition, they trained 3 different architectures and used their fusion. Their 3 models are very different from each other. Although performances of the models are the same as ours, their decision fusion boosted the overall score better than our decision fusion because different models complete each other better.

Third submission used the 2D CNN model [71]. They extracted 9 patches from different views. They used a different data augmentation technique (1 mm translation

in each axis and scaling the patches). In addition, they applied the same data augmentation technique to the test set. Final prediction for each candidate was obtained by averaging the probability of augmented data. This method improved the performance of their model.

Although our main aim was not to improve the overall accuracy, proposed model architecture is still comparable to many submissions. With deeper and different network structures, overall performance can be improved. Future work that can be done on this study is explained in the Conclusion chapter.

CHAPTER 5

CONCLUSION

The aim of this thesis was to propose a false positive reduction system for candidates obtained from lung CT images and compare the effects of different patch sizes when training the algorithm in order to increase overall performance. Candidate detection and classifying the candidates from lung CT images are very important for radiologists because analyzing these scans is a heavy burden for them. For a single scan, there are many slices to be reviewed. Reviewing these slices is both time consuming and open to oversight. Therefore, in order to get a robust result on these scans, sometimes, they are reviewed by more than one radiologists. Assisting the radiologists by reducing the number of candidates to review is a very important remedy. As the number of lung CT scans increased over the last year due to being more powerful in lung cancer detection, there is a need for a CAD system more than ever.

The proposed approach mainly consists of three steps: pre-processing, classification and decision fusion. In the pre-processing step, we carefully examined the scans. We have looked for how many slices exists for a scan, the distances between two consecutive slices, and the real world dimensions between two voxels on the same plane. We have found that, due to different scan machines and protocols, there was not a standard value in any of the parameters. Slice numbers vary from 120 to 600 while the distance between voxels changes from 0.4 mm to 1 mm. Yet, in order to get a more robust result in the classification step, these scans must be rescaled so that distance between two consecutive voxels must be the same in real world dimensions. After rescaling all scans, different patch sizes were determined and trained through the convolutional neural networks. In this step, we wanted to see how different volumetric

patches affect the training and testing performances. In total, 5 different patches were determined and they have been trained through a similar network. During the training, there was an important issue regarding the number of candidates in each class. In total there were 754,975 candidates but only 1557 of them were labeled as true positives. If the training was done in a full batch, we would see that models did not learn the structures of the true positives because true positives would be rarely trained by the network. In order to overcome this issue, modified mini-batch training was used in the training where the number of false positives and true positives were equal to each other. After the training of 5 patches, it has been observed that each patch generated a different result. When the smallest and biggest patches were compared there was a 0.107 point difference in their scores. In order to use strengths of different patches, ensemble of classifiers was used. We observed that fusion mechanism increases the best result by 0.025 points. Performance of the proposed framework in the thesis is promising for future development.

In the training step, two consecutive convolutional and max pooling layers were used. In order to learn internal structures of the nodules better, more convolutional layers can be added. One of the algorithms in this challenge used 7 convolutional layers. Training these 3D networks requires both time and GPU power. If these requirements are satisfied, more complex architectures can be experimented. Besides, 3x5x5 filter size was used in the convolutional layer, by using or adding bigger size like 7x7x7 or using smaller size filters like 1x1x1 consecutively, performance may be increased because these filters may capture bigger or specific features on the nodules. In addition, different network architectures such as inception modules can be used to improve performance.

Another factor that increases the accuracy of the detection algorithm was data augmentation. It is a widely used technique in the CNN algorithms. Data augmentation generates new data from original true positives so that network does not memorize certain patterns and increases its generalization ability. In the proposed method, we used rotation and mirroring properties. Zoom-in and zoom-out, shifting the nodule in different axes for a number of pixels have also been used as data augmentation techniques in literature [69] [71]. These data augmentation techniques, also, should be

done in the future work in order to improve the generalization ability of the CNN model.

One of the insights gained during this thesis is that 3D CNNs are very powerful in 3 dimensional object detection systems. When compared to the algorithms in ANODE09 study, even a simple 3D CNN framework outperforms the most complex clustering techniques. Yet, CNNs requires large number of training data. Without large number of samples, network only memorizes certain patterns and cannot get successful results on the test data.

Another insight that we gained from the experiments is that the receptive field of the input patch plays an important role in the training system. Small and large volumes focus on different characteristics on the training and ensemble of classifiers gives better result compared to each individual classifier. Therefore, experiments with different patch sizes must be conducted in these kinds of challenges.

To improve the performance of nodule detection system for clinical use, following points are planned to be used in the future work:

1. Using different network architectures such as Inception modules [43] or Capsule networks [80] in the CNN architecture.
2. Using more convolutional filters that can be used in order to detect complex features.
3. Employing different data augmentation techniques such as shifting and random zooming.

REFERENCES

[1] World Health Organization, World Cancer Report 2014, Lyon: International Agency For Research on Cancer, 2014.

[2] The National Lung Screening Trial Research Team, "Reduced Lung-Cancer Mortality with Low-Dose Computed Tomographic Screening," *The New England Journal of Medicine,* vol. 365, no. 5, 4 August 2011.

[3] S. G. Armato, M. L. Giger and H. MacMahon, "Automated detection of lung nodules in CT scans: preliminary results," *Medical Physics,* vol. 28, no. 8, pp. 1552-1561, 2001.

[4] H. Arimura, S. Katsuragawa, K. Suzuki , F. Li, J. Shiraishi, S. Sone and K. Doi, "Computerized Scheme for Automated Detection of Lung Nodules in Low-Dose Computed Tomography Images for Lung Cancer Screening," *Academic Radiology,* vol. 11, pp. 617-629, 2004.

[5] K. T. Bae, J.-S. Kim, Y.-H. Na, K. G. Kim and J.-H. Kim, "Pulmonary Nodules: Automated Detection on CT Images with Morphologic Matching Algorithm," *Radiology,* vol. 236, pp. 286-294, 2005.

[6] R. Belloti, F. De Carlo, G. Gargano, S. Tangaro, D. Cascio, E. Catanzariti, P. Cerello, S. Cheran, P. Delogu, I. De Mitri, C. Fulcheri, D. Grosso, A. Retico, S. Squarcia, E. Tommasi and B. Golosio, "A CAD system for nodule detection in low-dose lung CTs based on region growing and a new active contour model," *Medical Physics,* vol. 34, no. 12, pp. 4901-4910, 2007.

[7] J. Dehmeshki, X. Ye, X. Lin, M. Valdivieso and H. Amin, "Automated detection of lung nodules in CT images using shape-based genetic algorithm," *Computerized Medical Imaging and Graphics,* vol. 31, no. 6, pp. 408-417, 2007.

[8] M. N. Gurcan, B. Sahiner, N. Petrick, H. Chan, E. A. Kazerooni, P. N. Cascade and L. Hadjiiski, "Lung Nodule detection on thoracic computed tomography images: Preliminary evaluation of a computer-aided diagnosis system," *Medical Physics,* vol. 29, no. 11, pp. 2552-2558, 2002.

[9] Z. Ge, B. Sahiner, H.-P. Chan, L. M. Hadjiiski, P. N. Cascade, N. Bogot, E. A. Kazerooni, J. Wei and C. Zhou, "Computer-aided detection of lung nodules: False positive reduction using a 3D gradient field method and 3D ellipsoid fitting," *Medical Physics,* vol. 32, no. 8, pp. 1443-2454, 2005.

[10] B. van Ginneken, S. Armato, B. Hoop, S. Amelsvoort, T. Duindam, M. Niemeijer, K. Murphy, A. Schilham, M. Evelina, M. E. Fantacci, N. Camarlinghi, F. Bagagli, I. Gori, T. Hara, H. Fujita, G. Gargano, R. Bellotti, S. Tangaro, L. Bolanos, F. Carlo, P. Cerello, S. C. Cheran, E. L. Torres and M. Prokop, "Comparing and combining algorithms for computer-aided detection of pulmonary nodules in computed tomography scans: The ANODE09 study," *Medical Image Analysis,* vol. 14, pp. 707-722, 2010.

[11] A. A. A. Setio, A. Traverso, T. de Bel, M. S. N. Berens, C. van denBogaard, P. Cerello, H. Chen, Q. Dou and M. E. Fantacci, "Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: the LUNA16 challenge," 2017. [Online]. Available: https://arxiv.org/abs/1612.08012.

[12] A. Bhandare, M. Bhide, P. Gokhale and R. Chandavarkar, "Applcations of Convolutional Neural Networks," *International Journal of Computer Science and Information Technologies,* vol. 7, no. 5, pp. 2206-2215, 2016.

[13] K. Sharma and B. Preet, "Classification of mammogram images by using CNN Classifier," in *Advances in Computing, Communications and Informatics (ICACCI) International Coference on* , Jaipur, 2016.

[14] S. Liu, H. Zheng, Y. Feng and W. Li, "Prostate Cancer Diagnosis using Deep Learning with 3D Multiparametric MRI," [Online]. Available: https://arxiv.org/abs/1703.04078. [Accessed May 2017].

[15] "LUNA16 Challenge," [Online]. Available: https://luna16.grand-challenge.org/. [Accessed October 2017].

[16] "LUNA16 Challenge Results," [Online]. Available: https://luna16.grand-challenge.org/results/. [Accessed July 2017].

[17] "Non-Small Cell Lung Cancer Treatment (PDQ®)," [Online]. Available: https://www.cancer.gov/types/lung/patient/small-cell-lung-treatment-pdq. [Accessed January 2017].

[18] National Cancer Institute, "Non-small Cell Lung Cancer Treatment - Patient Version," [Online]. Available: https://www.cancer.gov/types/lung/patient/non-small-cell-lung-treatment-pdq. [Accessed 14 December 2016].

[19] American Cancer Society, "Lung Cancer," [Online]. Available: https://www.cancer.org/cancer/lung-cancer.html. [Accessed 14 December 2016].

[20] International Agency for Research on Cancer, "Cancer Fact sheets," [Online]. Available: http://globocan.iarc.fr/Pages/fact_sheets_cancer.aspx. [Accessed 20 December 2016].

[21] American Cancer Society Medical and Editorial Content Team, "Non-Small Cell Lung Cancer Survival Rates, by Stage," [Online]. Available: https://www.cancer.org/cancer/non-small-cell-lung-cancer/detection-diagnosis-staging/survival-rates.html. [Accessed 20 December 2016].

[22] American Association for Cancer Research, "AACR Cancer Progress Report 2012," 2012.

[23] "The Great Depression," [Online]. Available: http://www.history.com/topics/great-depression. [Accessed June 2017].

[24] American Cancer Society, "Can Lung Cancer Be Found Early?," [Online]. Available: https://www.cancer.org/cancer/lung-cancer/prevention-and-early-detection/early-detection.html. [Accessed 25 December 2016].

[25] S. Matsumoto, H. L. Kundel, J. C. Gee, W. B. Gefter and H. Hatabu, "Pulmonary nodule detection in CT images with quantized convergence index filter," *Medical Image Analysis,* vol. 10, no. 3, pp. 343-352, 2006.

[26] K. Murphy, B. van Ginneken, A. M. Schilham, B. J. Hoo, H. A. Gietama and 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,* no. 13, pp. 757-770, 2009.

[27] J. J. Koenderinkk, Solid shape, Cambridge, MA: MIT Press, 1990.

[28] F. Rosenblatt, "The Perceptron: A Probabilistic graphical model for information storage and organization in the brain.," *Psychological Review,* vol. 65, no. 6, pp. 65-386, 1958.

[29] S. Raschka, "Single-Layer Neural Networks and Gradient Descent," 24 March 2015. [Online]. Available: http://sebastianraschka.com/Articles/2015_singlelayer_neurons.html. [Accessed July 2017].

[30] M. Minsky and S. Papert, Perceptrons. An Introduction to Computational Geometry, Cambridge: MIT Press, 1969.

[31] S. Grossberg, "Contour Enhancement, Short Term Memory, and Constancies in Reverberating Neural Networks," *Studies in Applied Mathematics,* vol. 52, no. 3, pp. 213-257, 1973.

[32] F. Fukushima, "Neocognitron: A self-organized neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics,* vol. 36, no. 4, pp. 193-202, 1980.

[33] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning Representations by back-propagating errors," *Nature,* vol. 323, pp. 533-536, 1986.

[34] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation,* vol. 1, no. 4, pp. 541-551, 1989.

[35] Y. Bengio, P. Simard and P. Frasconi, "Learning Long-Term Dependencies with Gradient is Difficult," *IEEE Transactions on Neural Networks,* vol. 5, no. 2, pp. 157-166, 1994.

[36] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[37] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[38] G. E. Hinton, S. Osindero and Y. W. Teh, "A fast learning algorithm for deep belief nets.," *Neural Computation,* vol. 18, no. 7, pp. 1527-1554, 2006.

[39] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing System*, 2012.

[40] "IMAGENET Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)," 2012. [Online]. Available: http://www.image-net.org/challenges/LSVRC/2012/. [Accessed 2017].

[41] Khan Academy, "Overview of neuron structure and function," [Online]. Available: https://www.khanacademy.org/science/biology/human-biology/neuron-nervous-system/a/overview-of-neuron-structure-and-function. [Accessed 4 March 2017].

[42] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain," *Frontiers in Human Neuroscience,* vol. 3, November 2009.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," 2014. [Online]. Available: https://arxiv.org/pdf/1409.4842.pdf.

[44] A. Karpathy, "Stanford CS231 Class," [Online]. Available: http://cs231n.github.io/. [Accessed 3 November 2016].

[45] "Principles of training multi-layer neural network using backpropagation," [Online]. Available:

http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html. [Accessed November 2016].

[46] J. Duchi, E. Hazan and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.," *Journal of Machine Learning Reearch,* no. 12, pp. 2121-2159, 2011.

[47] Geoffrey Hinton, "Overview of mini-batch gradient descent," [Online]. Available: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. [Accessed 9 November 2016].

[48] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015.

[49] L. Bottou, "Stochastic Gradient Descent Learning in Neural Networks," AT&T Bell Laboratories, [Online]. Available: http://leon.bottou.org/publications/pdf/nimes-1991.pdf. [Accessed April 2017].

[50] "Early Stopping," [Online]. Available: https://deeplearning4j.org/earlystopping. [Accessed 6 January 2017].

[51] N. Srivastava, G. Hinton, A. Kreizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research,* no. 15, pp. 1929-1958, 2014.

[52] "Feature Extraction using convolution," [Online]. Available: http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution. [Accessed February 2017].

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

[54] IMAGENET, "IMAGENET Large Scale Visual Recognition Challange 2016 (ILSVRC2016)," [Online]. Available: http://image-net.org/challenges/LSVRC/2016/index. [Accessed March 2017].

[55] IMAGENET, "IMAGENET Large Scale Visual Recognition Challenge 2014 (ILSVRC2014)," 2014. [Online]. Available: http://image-net.org/challenges/LSVRC/2014/results. [Accessed March 2017].

[56] "Cancer Imaging Archive," [Online]. Available: https://wiki.cancerimagingarchive.net/display/Public/LIDC-IDRI. [Accessed September 2016].

[57] S. G. Armato, G. McLennan, L. Bidaut, M. F. McNitt-Gray, C. R. Meyer, A. P. Reeves, B. Zhao, D. R. Aberle, C. R. Henschke and E. A. Hoffman, "The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): a completed reference database of lung nodules on CT scans.," *Medcal Physics,* vol. 38, no. 2, pp. 915-931, 2011.

[58] C. Jacobs, E. M. Rikxoort, T. Twellmann, E. T. Scholten, P. A. de Jong, J. M. Kuhnigk, M. Oudkerk, H. J. de Koning, M. Prokop, C. Shaefer-Prokop and B. van Ginneken , "Automatic detection of subsolid pulmonary nodules in thoracic computed tomography images," *Medical Image Analysis,* vol. 18, pp. 374-384, 2014.

[59] A. A. A. Setio, C. Jacobs, J. Gelderblom and B. van Ginneken, "Automatic detection of large pulmonary solid nodules in thoracic CT images," *Medical Physics,* vol. 42, pp. 5642-5653, 2015.

[60] M. Tan, R. Deklerck, B. Jansen, M. Bister and J. Cornelis, "A novel computed-aided lung nodule detection system for CT images," *Medical Physics,* vol. 38, pp. 5630-5645, 2011.

[61] E. L. Torres, E. Fiorina, F. Pennazio, C. Peroni, M. Saletta, N. Camarlinghi, M. E. Fantacci and P. Cerello, "Large scale validation of the M5L lung CAD on heterogeneous CT dataset.," *Medical Physics,* vol. 42, pp. 1477-1489, 2015.

[62] O. Ronneberger, F. Philipp and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015.

[63] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," 2016. [Online]. Available: https://arxiv.org/abs/1605.07146.

[64] G. Xavier and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

[65] P. Cerello, S. C. Cheran, S. Bagnasco, R. Bellotti, L. Bolanos, E. Catanzariti and G. de Nunzio, "3-D Object Segmentation Using Ant Colonies," *Pattern Recognition,* vol. 43, pp. 1476-1490, 2010.

[66] D. R. Chialvo and M. M. Millonas, "How swarms build cognitive maps, in: The biology and technology of intelligent autonomous agents," Springer, 1995.

[67] Q. Li, S. Sone and K. Doi, "Selective enhancement filters for nodules, vessels, and airway walls in two- and three-dimensional CT scans," *Medical Physics,* vol. 30, pp. 2040-2051, 2003.

[68] A. Retico, P. Delogu, M. E. Fantacci, I. Gori and M. A. Preite, "Lung nodule detection in low-dose and thin-slice computed tomography.," *Computers in Biology and Medicine,* vol. 38, pp. 525-534, 2008.

[69] D. Qi, C. Hao, Y. Lequan, Q. Jing and H. Pheng-Ann, "Multilevel Contextual 3-D CNNs for False Positive Reduction in Pulmonary Nodule Detection," *IEEE Transactions on biomedical Engineering,* vol. 64, no. 7, pp. 1558-1567, 2017.

[70] "Theano - Machine Learning Library," [Online]. Available: http://deeplearning.net/software/theano/. [Accessed May 2017].

[71] A. A. A. Setio, F. Ciompi, G. Litjens, P. Gerke, C. Jacobs, S. J. van Riel and M. M. W. Wille, "Pulmonary Nodule Detection in CT Images: False Positive Reduction Using Multi-View Convolutional Networks," *IEEE Transactions on Medical Imaging,* vol. 35, no. 5, pp. 1160-1169, 2016.

[72] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar and L. Fei-Fei, "Large-Scale Video Classification with Convolutional Neural Networks," in

2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, 2014.

[73] K. He, Z. Zhang, S. Ren and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.," in *IEEE International Conference on Computer Vision*, 2015.

[74] "NumPy - scientific computing library for Python," [Online]. Available: http://www.numpy.org/. [Accessed March 2017].

[75] D. Scherer, A. Müller and S. Behnke, "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition.," in *International conference on Artificial Neural Networks (ICANN)*, Thessaloniki, 2010.

[76] K. G. Sheela and S. N. Deepa, "Review on Methods to Fix Number of Hidden Neurons in Neural Networks," *Mathematical Problems in Engineering,* 2013.

[77] "Keras: The Python Deep Learning Library," [Online]. Available: https://keras.io/. [Accessed September 2017].

[78] A. Dobrenkii, R. Kuleev, A. Khan, A. R. Rivera and A. M. Khattak, "Large residual multiple view 3D CNN for false positive reduction in pulmonary nodule detection," in *Computational Intelligence in Bioinformatics and Computational Biology*, Machester, 2017.

[79] "Ping An Technology," [Online]. Available: http://tech.pingan.com/en/technology.shtml. [Accessed December 2017].

[80] S. Sabour, N. Frosst and G. Hinton, "Dynamic Routing Between Capsules.," in *Neural Information Processing Systems*, Long Beach, CA, 2017.

[81] "Clarifai," [Online]. Available: https://www.clarifai.com/technology. [Accessed 27 June 2017].

[82] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," *Nature,* no. 323, pp. 533-536, 1986.