UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Lung Nodule Detection in Computed Tomography Scans Using Deep Learning

Author:
Mariia DOBKO

Supervisor:
Dr. Jan KYBIC

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

I, Mariia DOBKO, declare that this thesis titled, "Lung Nodule Detection in Computed Tomography Scans Using Deep Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Most advances in science come when a person for one reason or another is forced to change fields."

Peter Borden

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Lung Nodule Detection in Computed Tomography Scans Using Deep Learning

by Mariia DOBKO

Abstract

Accurate nodule detection in computed tomography (CT) scans is an essential step in the early diagnosis of lung cancer. Radiologists often use Computer-aided detection (CAD) systems to receive a second opinion during images examination. Nodule classification is a crucial stage of the full process, which comes as the second phase in a CAD system, right after candidates detection. Its task is to distinguish between true nodules and false positives.

The main goal of this thesis was to compare different deep learning methods, that can be used for nodule classification by evaluating their efficiency on a common database - LIDC-IDRI. We implemented three neural networks with 2-D convolution and three with 3-D, tested their performance and reported competitive FROC sensitivity scores. Used methods are compared among themselves and across other studies. Experimental results demonstrate a strong dependence between higher scores and 3-D CNNs application. For instance, VGGNet-11 gives 72.1% sensitivity at 8 FPs/scan, while same model with three dimensional convolution - VGGNet-11 3-D produces 91.9% at 8 FPs/scan rate. Based on the obtained results we recommend to use VGGNet-11 3-D for nodule detection, as it showed the best performance compared to other implemented methods. Moreover, received sensitivity of 91.9% at 8 FPs/scan and 90.6% at 4 FPs/scan rate demonstrates the promise of chosen network and its competitiveness with the state of the art method, which reported 92.2% at 8 FPs/scan and 90.7% at 4 FPs/scan. Our source code [1] is publicly available so it can be used for future work in other studies.

[1]https://github.com/MarichkaS/Lung_Nodule_Classification

Acknowledgements

I would like to express my sincere gratitude to Prof. Jan Kybic from the Czech Technical University (CTU) for his guidance in directing me throughout this thesis, generating new ideas, and sharing his knowledge. I want also to thank the Center for Machine Perception at CTU for the provision of necessary computational resources and the Faculty of Electrical Engineering of the CTU for an offer of the research scholarship. I appreciate support provided by Oles Dobosevych from the Ukrainian Catholic University at the toughest stages of this project as well as his useful and valuable feedback.

I owe big thanks to Dmytro Fishman from University of Tartu who introduced me to the medical imaging domain three years ago and got me interested in Data Science in general. Finally, I want to thank the Ukrainian Catholic University and the UCU Faculty of Applied Sciences for providing an excellent opportunity to study in a high quality academic program which has played an enormous role in shaping my professional interests and choosing a right direction of career development.

Contents

List of Figures

List of Tables

List of Abbreviations

CT	**C**omputer **T**omography
CAD	**C**omputer **A**ided **D**iagnosis
CNN	**C**onvolutional **N**eural **N**etwork
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
MIL	**M**ultiple **I**nstance **L**earning
LUNA16	**LU**ng **N**odule **A**nalysis 20**16**
LIDC-IDRI	Lung **I**mage **D**atabase **C**onsortium - IDRI
ROI	**R**egion **O**f **I**nterest

Chapter 1

Introduction

1.1 Context

Cancer has been a major cause of mortality for many years. One of its most common types worldwide is the lung cancer [34]. While being the most frequently fatal form of the disease, lung cancer can be prevented. Computed tomography has shown the survival benefit in annual screening, generally targeting individuals in a high-risk category. It is possible to find cancer at early stages when performing CT screening, and therefore, to decrease mortality caused by the disease [18]. The important role in increasing the quality and cost-effectiveness of a lung cancer diagnosis is assigned to Computer-Assisted Diagnosis algorithms. It performs nodule evaluation giving structured reports on their volume, localization, and other suggestions for further diagnosis and treatment [18]. These methods use Computer Vision techniques in order to extract useful information from each scan providing doctors with an automated nodule detection system. Nonetheless, it is a doctor who inspects the information received from the algorithm and uses his expertise to give a final conclusion about potential danger of a nodule.

1.2 Problem

No matter how advanced CAD systems are nowadays, they still can not produce results without any error. Often they are highly sensitive to such non-nodule structures like blood vessels, which results in a big number of false positive predictions. One of the steps in nodule detection task is to classify all the regions, which were received as candidates during ROIs extraction in order to decrease the number of non-nodules before showing all the candidates to a doctor. The difference in proportion of negatives to positives among them is very large: it is always the case, that detection algorithms find many regions of interest while just a small number of them actually contain a nodule. To increase the accuracy of a full end-to-end detection system, separate models are used for classification, which learn to distinguish between nodules and non-nodules. The received probability for each ROI, then, is used to filter out all the regions which certainly do not enclose any nodules, leaving a smaller amount of images for a doctor to examine.

Many techniques have been used for the classification step. Some approaches perform feature extraction using classical computer vision approaches followed by any kind of machine learning classifier (decision trees, k-Nearest Neighbor, Support Vector Machine, artificial neural network etc.). The others use end-to-end deep neural networks which complete feature extraction using convolution operation. The domain of medical imaging differs from other computer vision directions due to a specific data complexity that is present in many problems. However, it is common for the tasks involving medical images to use methods and neural networks, which were initially designed for other domains. In our task, datasets (scans) are three dimensional, giving a chance to use 3-D convolutional networks to solve nodule classification problem. Moreover, not using the whole volumetric region of the data, but representing it in 2-D will cause the lost of information about the whole candidate.

With a growth and evolution of deep learning many new model architectures appear. It is, thus, important to evaluate which of them give best performance in terms of the number of produced false positives, as well as time consumption. Sometimes, very deep neural networks are actually an overkill for a particular task, while usage of the less complex methods can preserve similar outcome, in the same time, save some resources. Each specific problem, hence, needs the complete and detailed studies, which would compare different approaches and provide a report on their effectiveness.

1.3 Data resources

LUNA16 is a competition held in 2016. The goal of this challenge was to compare different automatic nodule detection systems on a publicly available data set which includes CT scans, annotations, and a list of candidate regions for the nodules produced by organizers' algorithms. As it is stated in the rules of competition, LUNA16 focuses on a large-scale evaluation of automatic nodule detection algorithms on the LIDC-IDRI dataset [23].

LIDC-IDRI contains diagnostic and lung cancer screening thoracic computed tomography scans with annotated lesions. Seven academic centers and eight medical imaging companies collaborated to create this data set. It contains 1018 cases: 7371 lesions marked "nodule" by at least one radiologist, 2669 of these lesions were marked "nodule > or =3 mm" by at least one radiologist [23]. This makes it the largest publicly available database of CT scans.

Basically, the data set for LUNA challenge consists of images from LIDC-IDRI database, however, not every annotation from the latest is present in competition. In particular, non-nodules, nodules < 3 mm, and nodules annotated by only 1 or 2 radiologists are considered irrelevant and thus, are not included[23]. The organizers of the challenge extracted 1,186 lung nodules from LIDC-IDRI images and formed a data set from these nodules as positive candidates.

1.4 Our approach

In this thesis we apply several two dimensional neural networks to the annotated slices of ROIs extracted from CT scans. Those include a two-layered network based on [22], LeNet, VGGNet. Some of them are more complex (deeper), others have simpler architectures. We compare their performance among themselves, and then expand and implement all of the used models with 3-D instead of 2-D convolution. This allows us to measure how much dimensional representation of the candidates affects the results, and also how big is the role of number of layers in nodule classification. Evaluation using FROC analysis on predictions for LUNA16 dataset (consists of scans from LIDC-IDRI) showed the competitiveness of the used methods with other studies in false-positives reduction.

1.5 Goals

1. Provide an overview of previous works and achievements on lung nodules detection.
2. Apply different two and three-dimensional deep learning methods to the unified database of CT scans to perform classification of the regions which potentially contain nodules.
3. Evaluate and compare used neural networks among themselves, as well as with other studies from the literature.

1.6 Thesis structure

Chapter 2. Medical background
This chapter contains some background information on medical side of the nodule detection problem. It describes the importance of this task, defines nodules, explains Computer Tomography screening.

Chapter 3. Background
In this chapter, we specify the machine learning background needed for this project. In particular, we illustrate what is deep learning and how convolutional networks work.

Chapter 4. Related work
Here we analyze the previous works and research conducted in the area of our problem. We also depict the stages of nodule detection.

Chapter 5. Methods
This chapter describes the approaches, which we used for ROIs classification, in other words, for false-positives reduction step.

Chapter 6. Implementation
Here, the explanation of the full process of implementation is specified in details. We include the link to the source code and mention briefly the functionality of each module.

Chapter 7. Experiments
In this chapter, we do a careful evaluation of used methods using different dataset splits and two numeric metrics: overall accuracy, and FROC sensitivity. We report those results and compare them to the scores from others works submitted to LUNA16 challenge.

Chapter 8. Conclusions
We summarize the achieved results and their comparison. In the same chapter we describe the ideas for future work.

Chapter 2

Medical background

2.1 Lung cancer statistics

The leading killer among all forms of cancer is lung cancer. Its severity can be easily shown by the very low survival rate, as well as the number of estimated new cases worldwide and its tendency to progress quickly. What makes it even more alarming is the fact that symptoms do not usually occur until the cancer is advanced. The number of deaths caused by the lung and bronchial cancer exceeds all other types of cancer with over 153,000 cases in total per year in the United States. It is almost 26% of all other sites of cancer. Lung cancer is ranked second in the number of new cases in recent years. Meanwhile, it is one of the most dangerous types of cancer as its 5-year relative survival rate at all stages is only 18%: 15% for men and 21% for women.[1]. Survival time decreases significantly with the progression of the disease, so it is crucial to diagnose and treat cancer as soon as possible.

2.2 Risk factors

The dominant risk factor for lung cancer is cigarette smoking, it is a cause of more than 80% of lung cancer deaths in the US[1]. The quantity and duration of smoking are also important factors. Exposure to involuntary smoking increases lung cancer risk by about 25%, a finding replicated worldwide [13]. Other potential causes of lung cancer include the exposure to radon gas released from soil and building materials, occupational or environmental exposure to second-hand smoke, certain metals, some organic chemicals, radiation, air pollution, and diesel exhaust [1]. Sometimes, genetic predisposition can cause the development of lung cancer.

2.3 Nodules and tumors

Lung cancer tumors are of different types. Two major types are non-small cell lung cancer (NSCLC) and small cell lung cancer (SCLC). NSCLC accounts for about 85% of lung cancers. The basic characteristics analyzed during diagnosis are type and stage. The stage of the disease depends on whether the cancer is local or has spread from the lungs to the lymph nodes or even other organs. The staging system is different for non-small and small cell cancer. The treatment for these two types also differs, hence, it is important to understand that they are very distinct. The approaches to treatment include surgeries, chemotherapy, radiation, targeted drugs, immunotherapy. It is easier to deal with early-stage small tumors in lungs than late-stage cancer which has spread to other parts of the body. For this, usually, chemotherapy is used, alone or combined with radiation [1]. This approach provides remission for a large percentage of patients, but there are still big chances for the cancer to return. Therefore, detecting small nodules which can potentially be tumors on the first stages can help in early diagnosis.

Lung nodules can be cancerous, though in most cases they are benign. A larger lung nodule is more likely to be cancerous than a smaller one, but it's a doctor who is eligible to make a conclusion of how dangerous the particular nodule is. In many cases, it is helpful to see if a nodule changes or grows over time, thus different types of screening are often recommended.

2.4 Computed tomography screening

The detection and diagnosis of lung cancer have improved with the development of computed tomography (CT). Screening with low-dose spiral computed tomography (LDCT) has been shown to reduce lung cancer mortality by about 20% compared to standard chest x-ray[1]. The screening process involves performing tomography on the high-risk group, for example, current or former heavy smokers who seem to be in good health. With the use of CT, doctors are capable to spot the dangerous nodules in lungs when those are at the early stages providing enough time to perform the treatment. Over 80% of the 5-year survival rate has been reported in surgically treated peripheral lung cancer [33]. This same research reports that the advances in screening with CT enabled doctors to detect lung tumors smaller than 2 cm in diameter[33].

FIGURE 2.1: Examples of CT scans from LIDC-IDR data set [23]: images represent 2-D slices retrieved from the centers of separate scans

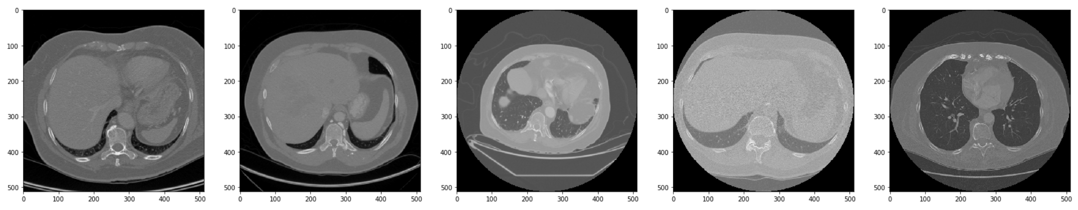
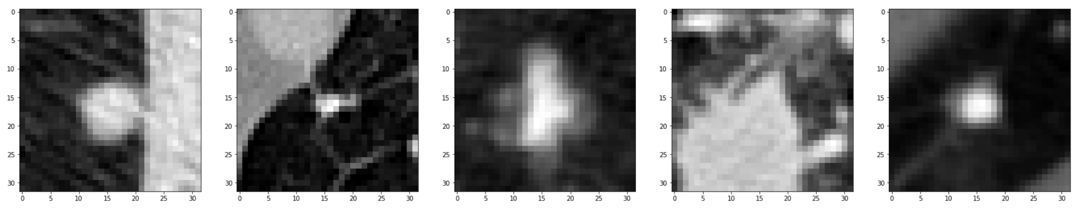


FIGURE 2.2: Nodules from the scans: images are 32x32 pixels regions containing a tumor. Some nodules are smaller, others can't fully fit into 32x32 size



Chapter 3

Background

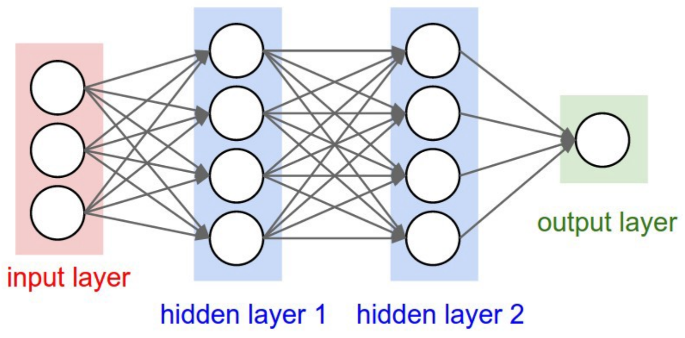
3.1 Neural networks

In 2017 according to PubMed neural networks were the most popular supervised learning technique in medical applications. They have been successfully used to diagnose/detect or predict cancer, Parkinson's disease, diabetes, diabetic retinopathy, osteoporosis, nerve disorders, Huntington and other diseases.

So what is a neural network? A neural network consists of some number of hidden layers with neurons at each of them. Neuron represents a mathematical function which translates its inputs using weights and an activation function into a single result and then sends it to another neuron in the next layer, see Figure 3.1.

An example of the neural network's architecture on Figure 3.1 is an acyclic graph. The outputs of some neurons are inputs to other neurons in the following layer (in a fully connected network two adjacent layers are fully pairwise connected), but there are no connections between neurons within a layer. This represents the most common layer type - fully-connected. However, there are plenty of other types of layers which extract or translate features from their inputs in different ways. Those include one, two and three-dimensional convolutional layers, recurrent layers, pooling layers, normalization layers, and other more specific ones.

FIGURE 3.1: Example of a 3-layer neural network with three inputs, 2 hidden layers of 4 neurons each and one output layer from [17]



To train the network means to estimate the best weights in all neurons to minimize the error between the outcome and correct predictions. The problem of training is equivalent to the problem of minimizing the loss function (function that measures the deviation between a predicted value and actual label). The algorithm which is used to optimize the loss function is chosen from the gradient-based algorithms, the simplest example is Stochastic Gradient Descent.

3.2 Convolutional neural networks and deep learning

As mentioned above, there exist several types of hidden layers, one of them is called convolutional. This type of layers has neurons arranged in three dimensions: width, height, and depth. It basically consists of a set of learnable filters.

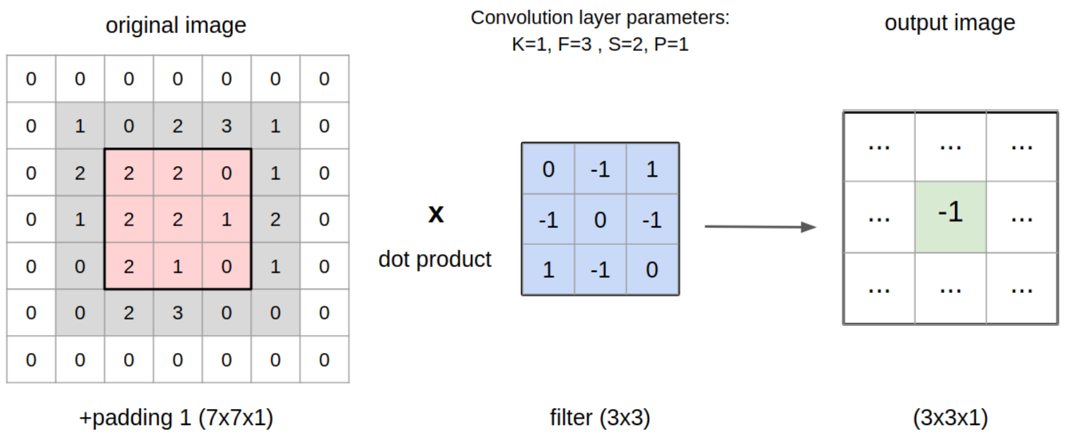
The typical filter has 3 channels corresponding to the RGB image. Its size is also called the receptive field and it defines the amount of a local region of the input image which is assigned to the neuron. When performing convolution we slide each filter across the image (input) vector and compute dot products between the entries of the filter and the input at any position. This process produces a 2-dimensional activation map. The set of the activation maps produced by all the filters is stacked along the third dimension (responsible for depth) and passed ahead to the next layer. The main idea is to learn filters that activate when they see some type of feature, for example, an edge, a blotch, or even some kind of pattern. The further (deeper) filter is situated in the network the more abstract and pattern-like feature it will be able to catch.

To sum up, the convolutional layer requires four hyperparameters: number of filters - K , receptive field - F , the stride - S (step of a filter), the amount of padding - P . As the input we have an image or vector of shape: $W \times H \times D$. Stride and padding control the size of the output volume: stride defines the size of the step for moving the filters, padding increases the input volume by surrounding it with certain numbers (usually zeros) around the border.

The output of the convolution operation has width equal to $(W - F + 2P)/S + 1$, while height equals to $(H - F + 2P)/S + 1$. The number of dimensions is the number of filters. The choice of hyperparameters is usually based on common conventions and rules of thumb.

Convolutional neural networks show very effective results in image and video detection, segmentation, localization, classification, natural language processing, and even recommender systems. Deep learning can be referred to as the extension of the classical neural network technique. Deep neural networks have many hidden layers hence a very large number of parameters. Deep convolutional networks have been commonly used in image classification tasks, reporting best performance even on the most popular benchmark dataset - ImageNet [29]. Deep learning can explore more complex non-linear patterns in the data.

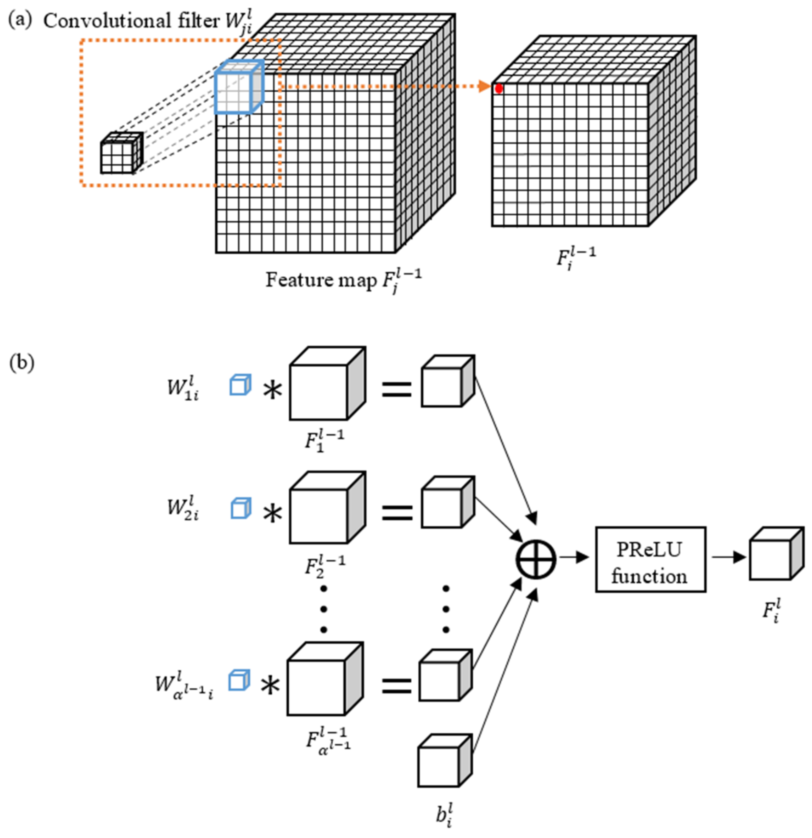
FIGURE 3.2: Example of convolution operation



3.3 3-D convolution

The main difference between 2-D and 3-D convolution is that the last one operates not on single 2-D images, but on the voxels (cubes extracted from 3-d image). The input to such a network requires the data to be presented in three-dimensional space. Thus, this type of layers are often used on videos or medical images, for example, computed tomography, X-ray, MRI scans. 3-D Convolution can be used to find patterns across 3 spatial dimensions. This approach is commonly used for action videos, because 3-D networks can be fed with multiple image frames which were concatenated across a temporal dimension, thus patterns are found across frames. The process of kernel sliding in 3-D convolution operation is similar to regular convolution: the kernel slides in 3 directions with a predefined stride and size of the kernel at every step calculating the dot, however, the output is now three dimensional.

FIGURE 3.3: Example of 3-D convolution.(a)3-D convolution of a feature map with a filter.(b)Generation of the i-th feature map (F) in the l-th layer src: https://doi.org/10.1371/journal.pone.0185844.g004



Chapter 4

Related work

The task of nodule detection has been approached by many researchers in recent years. It is a promising area of study. CAD (Computer Aided Detection) systems can help radiologists in performing a better detection of nodules as it can offer a second opinion on early diagnosis of lung cancer. Datasets of Computed Tomography (CT) scans: LUng Nodule Analysis 2016 [24], LIDC [23], ANODE09 [38], TIME [7] have been collected and annotated by specialists. Being publicly available they provide convenient data samples for training and testing proposed CAD systems. However, these datasets vary in number of patients, number of slices per scan, and scanning configurations like slice thickness or slice spacing. The main task for all of them remains the same: detect all the nodules present in the CT scan.

The process of solving this problem usually requires two steps: candidates detection and false-positive reduction (classification). This division proves to give better results, rather than one-step nodule detection. The candidates detection step provides the system with a large number of false positives. This happens due to their high sensitivity to such non-nodule structures like blood vessels, which they interpret as nodules. This is why the second stage is required. Overall, it is the classification of candidates (nodules) that is responsible for the final performance of the system. The evaluation is presented by the Free-Response Operating Characteristic (FROC) analysis [32].

The performance metric was previously introduced in the ANODE09 challenge and is used in LUNA16 challenge. It calculates the sensitivity of the average number of false positives per scan, called FP rate. The final score is the average of the sensitivity at seven false positive rates: 1/8, 1/4, 1/2, 1, 2, 4, and 8 FPs per scan. By measuring the metric score on each of the rates, FROC curve can be drawn. The 95% confidence interval is achieved using bootstrapping with 1,000 bootstraps.

This work is focused on testing and comparison of different approaches to the classification step through application of 2-D and 3-D Convolutional Neural Networks.

4.1 Nodule candidates detection

In this task both classical computer vision detectors and deep learning can be used to detect potential regions of nodules' position. Many techniques for this task were presented in the last years: application of hysteresis thresholding [26], usage of a double-threshold density mask within the lung regions for mask generation followed by morphological erosion and a connected component analysis in order to obtain clustered candidates [15], a three-dimensional lung segmentation algorithm along with a multistage process of thresholding and morphological operations [30], nodules' centers locations were found with nodule and vessel enhancement filters and a computed divergence feature in [35], Channeler Ant Model as a segmentation tool was used in [37].

When generating proposed candidates for LUNA16, the organizers combined different approaches. All the found ROIs were merged at the end and published for their further use in false-positive reduction step [30]. They also prove that combination of several candidate detection methods improves the sensitivity of the system comparing with the one algorithm (adapted from [35]) which showed the best performance in LUNA16 challenge among the single methods.

CAD system proposed by [6] claims to achieve even better results than the winner of LUNA16 challenge, it applies Faster Region-based Convolutional Neural Network (Faster R-CNN [28]) for candidate detection.

4.2 Candidate classification

The false-positive reduction step depends a lot on the results of the candidates detection because it operates on the found ROIs. Many approaches were proposed for this task:

- 2-D CNNs: [22], [5], [39]
- 3-D CNNs: [40], [12]
- Other classifiers: SVM - [3], feed forward neural network - [37], asymmetric AdaBoost - [9], novel classifier that evolves ANNs using genetic algorithms - [36]

Some methods extract features from the candidates regions with help of thresholding and morphological processing, then they apply statistical and boosting classifiers over this data. Others use deep learning and go either to two dimensional or three dimensional convolution, in the second case interpreting each sample as a voxel with a certain number of slices belonging to one example.

4.2.1 Standard classifiers

Approach presented in [9] obtained the candidates for classification by utilizing thresholding and morphological image processing. Features were hand-crafted from information about volume, shape and intensity [8]. It further applied asymmetric and symmetric AdaBoost classifiers and was tested on several databases: TIME, LIDC/IDRI and ANODE09. [20] used a binary decision tree as a classifier on autoencoder generated features. [3] used a SVM classifier for false-positive reduction. It finished third in LUNA16 challenge.

4.2.2 2-D Convolutional neural networks

Inspired by the great achievements of convolutional neural networks in computer vision, especially in ImageNet classification benchmark, this type of models was used for nodule detection. In [22] it was showed that even simple 2-D CNN layers can produce promising results. Model consisted from only two convolutional followed by downsampling and three fully connected layers with softmax at the output. This algorithm produced 87.1% sensitivity for 4.62 FP/example rate. [31] created multiple streams of 2-D ConvNets, for which the outputs are combined using a dedicated fusion method in order to get the final classification. [39] shows how to combine dedicated detection system and off-the-shelf CNN features to succeed in the task of false-positive reduction for nodule detection.

4.2.3 3-D Convolutional neural networks

The nodule detection in CT scans can be also approached as a 3-D object detection problem, thus, it makes sense to use models which can find and work with all the information present in three dimensional images. According to [12] the 3-D CNNs can encode richer spatial information and extract more representative features than 2-D convolutional networks. Several completed experiments reported the efficiency of using 3-D CNNs in medical imaging. It was previous usage of such networks that inspired the idea of applying this technique to this particular task [16], [10], [11], [41], [19]

For nodule detection problem [12] proposed an architecture with 3-D convolutional layers CUMedVis (name was given by authors) which achieved the highest performance metric score in LUNA16 challenge scoring 85.4% sensitivity at 1.0 false positive/subject rate. This framework was developed by joining three 3-D CNNs to produce final classification probabilites for each nodule candidate. Not one but three networks were used to include multilevel contextual information: each CNN has a different size of receptive fields. It is a receptive field which is responsible for the amount of surrounding contextual information near the target region to be included in training. The architecture consists of 3-D convolutional layers, 3-D max-pooling, and fully-connected layers for feature extraction, with a softmax in the end for obtaining probabilities.

There are other researches that explored 3-D networks and their potential efficiency. [40] proposed 3-D G-CNNs to solve false-positive reduction step. They claim having received high classification accuracy along with sensitivity and more efficiency in terms of performance.

Chapter 5

Methods

5.1 2-D CNN from Li et al.

For the first approach we used a convolutional neural network with two-dimensional convolution. Its architecture was proposed by [22]. The main idea is to use the operation of convolution several times to retrieve features followed by fully-connected layers and train the network to distinguish between two types of classes (nodule and non-nodule) predicting the probability for each of them with softmax function on the last layer. Two models are described below. Both networks might be compared with LeNet-5 architecture proposed in [21] because of the similarities between architectures. However, if one takes a closer look there are several differences: the number of feature maps on the first convolutional layer for second model (CNN T5), the number of neurons on each fully connected layer, the presence of additional dense layer in both: CNN T4 and CNN T5 approaches.

5.1.1 CNN T4

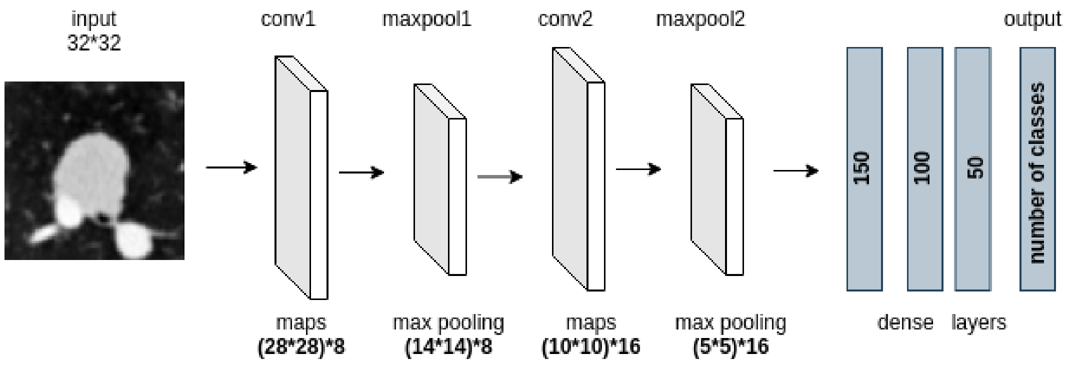
The network's structure depends largely on the input size of the images fed to it. The paper [22] submitted a deep CNN, which is constructed on 32 by 32 pixels image data. The network consists of two convolutional layers connected by a downsampling (max pooling) layer, which reduces the spatial size of its input and the amount of parameters, reducing the necessary computational power. Downsampling also helps to prevent overfitting. The first convolution contains 6 feature maps. The kernel size is 5 for every convolutional layer while the step (stride) of kernel is 1. For downsampling layers: the kernel size equals to 2 and the step (stride) is 2. The second convolution in this network has 16 feature maps and is followed by another max pooling layer, outputs of which are passed to a dense layer. In total, there are four fully connected layers with 150, 100, 50, and 2 nodes correspondingly. The last two nodes are responsible for the output probabilities of nodule and non-nodule. While we use convolutional layers with purpose of feature extraction, we need all the dense layers to perform classification task. Basically, fully connected layers learn non-linear combinations of the extracted features. This step makes the model end-to-end trainable.

5.1.2 CNN T5

For experimental purposes the same network is modified: the convolutional map size is set to 8 instead of 6. All the others parameters are kept the same, including the size of the input data. To prevent overfitting we added dropout regularization to CNN T4 and CNN T5, which was not present in the original architecture from [22]. We placed dropout layer between first and second dense layers and one more between second and third fully connected layers. The proportion of the input units to drop was set to 0.3.

The full architecture of this network is presented in Figure 5.1.

FIGURE 5.1: Architecture of deep CNN by [22]. The input data is ROI image pixels (1024-dimensional vector). The output consists of two neurons (nodule: 1 and non-nodule: 0) src: [22]

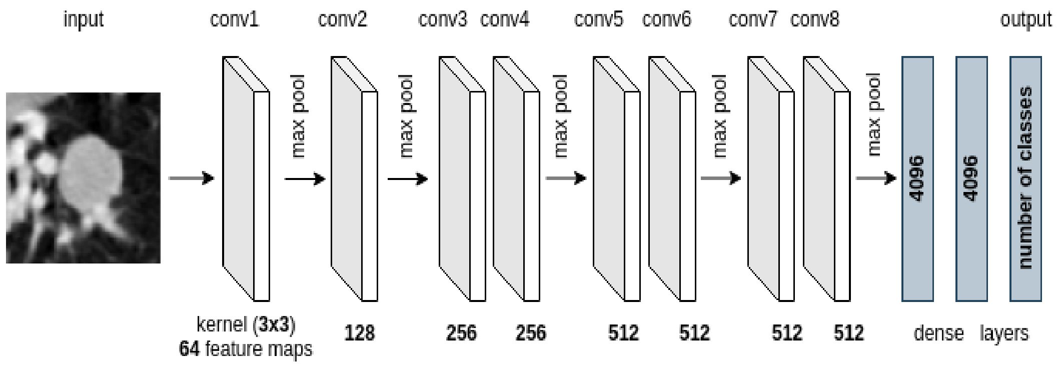


5.2 VGGNet

With the development of Deep Learning emerged the need of a unified baseline for measuring the efficiency of proposed frameworks. ImageNet ILSVRC challenge presented [29], which became the most popular benchmark dataset for this purpose. New convolutional neural networks architectures are usually tested on this benchmark which gives us an opportunity to compare their performance with each other.

In 2014 the network from Karen Simonyan and Andrew Zisserman known as the VGGNet was in top-2 best performing networks in ILSVRC challenge after obtaining the error rate under 10%. This model is frequently used for classification or feature extraction on different datasets. At the moment of its submission VGGNet contributed into Deep Learning by showing that the depth of the network is a crucial element for good performance. Nowadays there are many other models built on top of VGGNet or based on its idea of using a 33 convolution layes, and these models are applied to many domains.

FIGURE 5.2: VGG-11 network: 8 two dimensional convolution layers and 3 fully connected, the last dense layer is followed by softmax function for class prediction. The number of feature maps on each layer is mentioned under each of them.



There are several configurations to VGGNet. The most famous one contains 16 layers (including both: convolutional and fully connected layers). Other configurations vary in the number of layers and are named correspondingly: VGG-11, VGG-13, VGG-19. In this work we use VGG-11, its architecture is shown on Figure 5.2.

VGG architecture consists of 'blocks' which have two convolution layers. They apply same filter size - 3x3 2-D convolution with stride 1 and pad 1 multiple times in order to extract more representative and complex features. The size of pooling kernel remains the same on all layers and equals to 2x2 with stride 2 and no padding. The biggest issues with VGG usage is that it has a large number of parameters - over 130 millions, is expensive to evaluate, and uses a lot of memory.

5.3 3-D LeNet

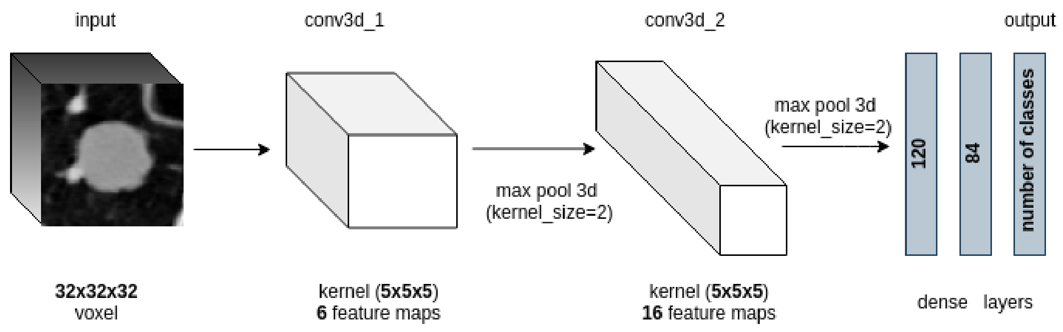
There is another characteristic of the data which can be useful and might influence the choice of the approach: LIDC images are obtained from CT scans, which are three dimensional in their nature. The nodules which the algorithm is trying to classify have three dimensional structure, meaning there is more important information in the whole voxel which contains the nodule rather only its one slice. Using 3-D neural network instead of 2-D in this case can produce better results as three dimensional convolution is able to retrieve features which are relevant for volumetric data. However, the downside of this technique is the need of massive computational power to train a 3-D model.

As it was mentioned above, the network proposed by [21] has similar architecture to [22], which achieves 0.86 sensitivity at 5 FP rate on LIDC dataset which is a good and competitive result. This means that, even though, LeNet has much less layers than VGG, it can be used as a baseline in similar computer vision tasks. The idea here is to improve the performance of [22] by expanding their network to train on three dimensional data.

Before this step, we train and evaluate 3-D LeNet to see if this relatively simple but computationally lighter (comparing to deep nets like VGGNet) network can produce meaningful results.

The structure of the model is shown on Figure 5.3.

FIGURE 5.3: 3-D LeNet: two 3-D convolutional layers with 5x5x5 kernel 6 feature maps on the 1st layer and 16 on 2nd, connected by 3-D maxpooling layer with kernel size equals 2 and stride 2, followed by three fully connected layers with 120, 84, and 2 neurons on each.



5.4 3-D CNN T5

As mentioned above, we want to improve classification results by using three dimensional convolution on volumetric voxels rather than working with 2-D slices. This approach has been already used in other studies, but in this work we provide a comparison of several architectures, each of which is implemented independently with 2-D and 3-D convolution to measure how much this adjustment can increase the metric scores. After using LeNet in 3-D, we change network from [22] by adapting it to learn three dimensional data. Changed network consists of two 3-D convolution layers each having 5x5x5 kernel size with no padding and stride 1. First convolution layer has 8 feature maps, while second one has 16. The number of neurons on last four fully connected layers remain the same as in [22] architecture: 150, 100, 50 and 2.

5.5 3-D VGGNet

We also transform convolutional layers of VGGNet-11 to three dimensions. Now we can test whether training two same architectures but with different convolution dimensionality would give very different results and if so, which configuration would perform better.

Chapter 6

Implementation

We divided the implementation into several stages. During this development we built separate modules, each responsible for a particular task. We made repository with the code[1] publicly available. Its short description is provided below:

1. **Data preprocessing** The module for operations on raw CT scan images. Those include data reading, extraction of candidate patches, regions saving, augmentation, train-test split and others. We wrote these functions with the idea to make their usage possible at all stages of this work. For example, the module responsible for loading the data can be used during training as well as during prediction. The code is located in folder *data_preprocessing* on our github.

   For instance, in our repository the folder named *data_preprocessing/load_tools.py* contains the following modules: load tools (helper functions for reading images, converting them to different system of coordinates, extracting the needed ROIs and saving them) and dataloaders pytorch (provides a variety of different data loaders for training any network on pytorch framework). There are also four ipython notebooks which show how to use mentioned modules for data augmentation, LUNA16 exploration, visualization, and regions saving.

2. **Modelling** For each CNN architecture we have written a module on Keras and a separate one on PyTorch. Therefore, it is possible to train models in different environments: either with installed Keras (on TensorFlow backend) or PyTorch, or both of them. This allows us to compare the frameworks' performance and be more flexible in the process of building new networks or changing the existing ones. Read more about this in 6.1.

   One can find implemented models in *src_keras/models_keras.py* and in *src_pytorch /models_pytorch.py* .

3. **Training and prediction** We created two different modules for training and testing one for Keras syntax and another for PyTorch. Both of them save the trained models and their weights to files, so the test prediction step can be launched separately from the training.

   In our code folders *src_keras* and *src_pytorch* contain all the necessary modules to train, save, and evaluate models implemented on both frameworks. For instance, module *src_pytorch/train_tools.py* combines all the functions needed for training a network, example of this process is shown in ipython notebook named *TrainCNNs.ipynb*. In the same time, we have *src_pytorch/test_tools.py* which provides similar functionality and allows to extract label predictions, details are described in *PredictCNNs.ipynb*. Modules *src_keras/train_tools.py* and

[1] https://github.com/MarichkaS/Lung_Nodule_Classification

src_keras/test_tools.py are presented for training and testing using keras. These processes can be followed through in *Train2dCNNs.ipynb*, *Train3dCNNs.ipynb*, *Predict2dCNNs.ipynb*, and *Predict3dCNNs.ipynb*.

4. **Evaluation** In this step we used the module provided by LUNA16 competition. It takes the csv file with predicted probabilities for each candidate and runs FROC analysis to report the overall sensitivity of a system at the ten predefined false positive rates. Code can be downloaded from a folder called "evaluation script" on the Data page of LUNA16 challenge[2].

   File *noduleCADEvaluationLUNA16.py* launches FROC analysis, the only things that are required to run it are the paths to csv files with annotations, names of scans, output directory, and predicted results for each region (should include scan name, x y z coordinates of the center of the region, and a probability of being a nodule).

6.1 Frameworks

We use PyTorch [27] and Keras [4] deep learning frameworks with integration into Python 3.6. Both of them are neural networks API which provide building blocks for developing deep learning models. Keras is a high-level library which requires a backend engine. From the three available backend implementations we chose TensorFlow backend for this work. Pytorch, on the other hand, shares some C++ backend with the deep learning framework Torch.

PyTorch is more complex than Keras because it operates on a lower level and requires more custom configurations. Nevertheless, it is best known for its flexibility, short training duration and debugging capabilities. In our case, processing a big dataset of 3-D images is very time consuming, so this framework is useful in decreasing the training time. Moreover, PyTorch gives an opportunity to add some custom changes to the networks, optimization, loss calculation providing a possibility to adjust the architecture for any experiment. The performance is comparatively slower in Keras, however, its code readability is more clear. We chose to build methods using Keras because it is a fast and convenient way to develop models and test a hypothesis. Furthermore, the written code can then be reusable by others due to its simplicity.

We implement each approach described in section 'Methods' using both frameworks separately, except for VGGNet which is already included in Pytorch and thus was used as an imported module.

6.2 Computational resources

In our experiments we train all the models on GeForce GTX 1080 Ti GPU which has 11,264 MB GDDR5X memory on the card.

For the models CNN T4 and CNN T5 100 iterations take about 22 seconds per iteration to train on 30,000 two dimensional images with size 32x32 pixels and batch size 64. With the same configurations one epoch of VGGNet-11 runs for approximately 42 seconds, which is twice longer than previous two networks. Computational time increases drastically when we use 3-D voxels. For example, on the same amount of training data VGGNet-11 3-D takes 320 seconds for one epoch.

[2]https://luna16.grand-challenge.org/Data/

Chapter 7

Experiments

7.1 Dataset

Dataset includes 888 CT scans from LIDC/IDRI database [23]. This means that from 1018 scans available in database selected were only those with a slice thickness smaller than 2.5 mm. All nodules (positive samples) are greater than or equal to 3 mm and were accepted by at least 3 out of 4 radiologists. The regions of interest for us are 3-D voxels containing nodules, this is why we need to cut out chunks from each scan. These extracted regions will be the input data for neural networks. To perform this step we take ROI candidates, which are proposed in LUNA16 challenge. For each candidate we receive information about its location (x, y, and z position) in world coordinates, and the annotation class. LUNA16 organizers claim in [32] that they obtained this data using three candidate detection algorithms [26] [15] [30].

All in total we have 551,065 candidates which include 1,186 true nodules.

The three dimensional candidate contains a lesion positioned in the center but it also has a lot of background around it. Since nodules can be of a different size, some slices consist mainly of background and usually only the center three-five slices have a nodule. Big lesions, on the other hand, can appear on many slices through Z plane. The largest nodule (Figure 7.3) is 32.27 mm in diameter while the tiniest is ten times smaller - 3.25 mm (Figure 7.1). Thus, labeling all slices from the candidate with the class given to its center is incorrect. To work with two dimensional data we need to annotate as positive class only those slices from candidates that contain a nodule. To be certain that no incorrectly labeled samples are present in our training dataset, we use only middle slices to represent a volumetric candidate in two dimensions. We have tried to use more slices from the voxel, but in many cases they appeared to not have a nodule even on just two slices away from the center. In Figure 7.2 we see that from the whole 3-D region only the middle 4 slices actually contain a nodule. Nodules' position in the lungs varies a lot: some are attached to tissues, some are surrounded by many artifacts, others have only dark background around them, for examples see Figure 7.4.

7.2 Preprocessing

According to [22] the nodules whose sizes are less than 32x32 pixels account for 95.33% of the data, and less than 64x64 pixels represent 99.991% of nodules. We created dataset which contains candidates of size 32x32x32, to be confident that using 3-D convolution on these images would fit into memmory limits.

Since there is a strong disproportion in the amount of positive to negative ROIs we use augmentation to form a balanced training dataset. For each candidate with a nodule we perform random rotations of the input by 90 degrees, transpose the

FIGURE 7.1: Region of size 48x48x48 with the smallest nodule from the dataset. Because the nodule is very small we can see it only on the central slices (the ones in the bounding box), zoomed in Figure 7.2

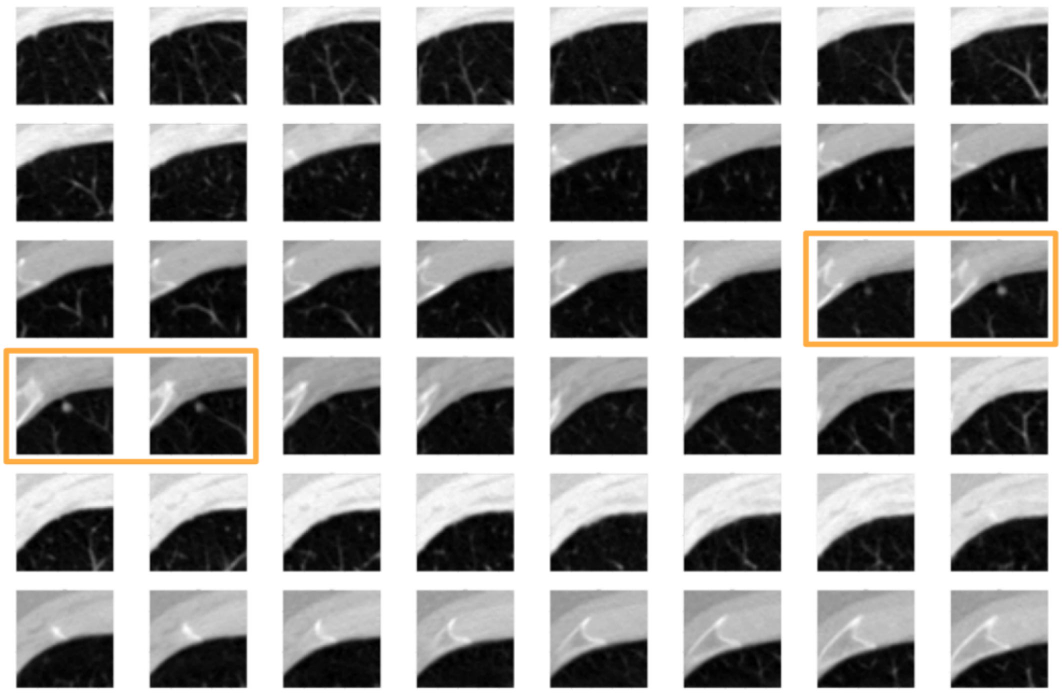


FIGURE 7.2: Center slices of the region voxel with the smallest nodule from the dataset. The whole ROI is shown in Figure 7.1

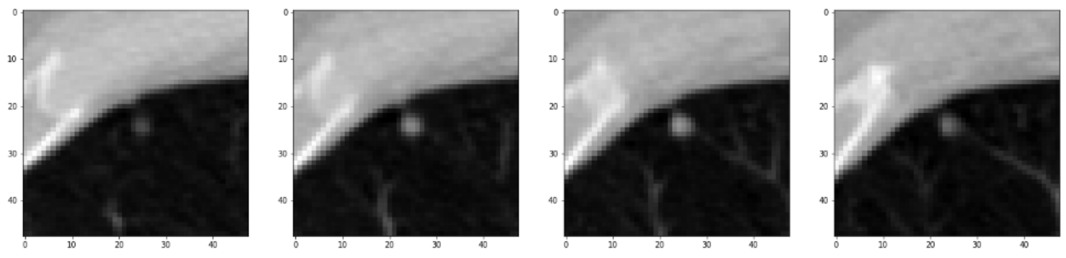


image by swapping rows and columns, adjust brightness and contrast, vertically and horizontally flip the input, and use scaling.

7.3 Train-test split

We have noticed that the overall performance is largely dependent on the proportion of classes. Because we work with a very unbalanced dataset, multiple configurations for splitting the data have been tested.

1. **D1 split**

   Before augmentation we select 15,000 non-nodules candidates for training and 8,000 for testing. To do this we group by scan all proposed candidates of non-nodule class, then randomly choose 17 samples from each group saving them for training and 9 samples for testing. This means that we receive 65% to 35% train to test ratio. Those regions were previously saved as 3-D voxels, so when 2-D input data is needed we take only one middle slice of size 32 by 32 pixels from each candidate. D1 split is the way we divide data and not the way we preprocess it, so 3-D D1 contains same images as 2-D D1 split, they differ only in dimensionality of the images.

For each out of 1,186 3-D nodules we perform image augmentation, which are listed above in Section 7.2, receiving around 20 generated samples from one candidate. After this step we obtained an upsampled class of positives - 23,000 three dimensional nodules in total. Those are then split to 15,000 regions for training and around 8,000 for testing to preserve 65% to 35% proportion. Next, we retrieve one middle slice from each to have 2-D images for training (same procedure as for non-nodules).

In conclusion, after all these operations we have a 30,000 images dataset for training and 16,000 samples for test. We calculate test accuracy and test loss during training to keep an eye on possibility of overfitting. In this thesis we refer to this dataset split configuration as D1. It consists of D1 testing set and D1 training set.

2. **D2 split**

   CT scans, in general, contain more background information rather than regions with nodules. This means that in real world we will always see more negatives

FIGURE 7.3: Candidate voxel of size 48x48x48 with the largest nodule from the dataset

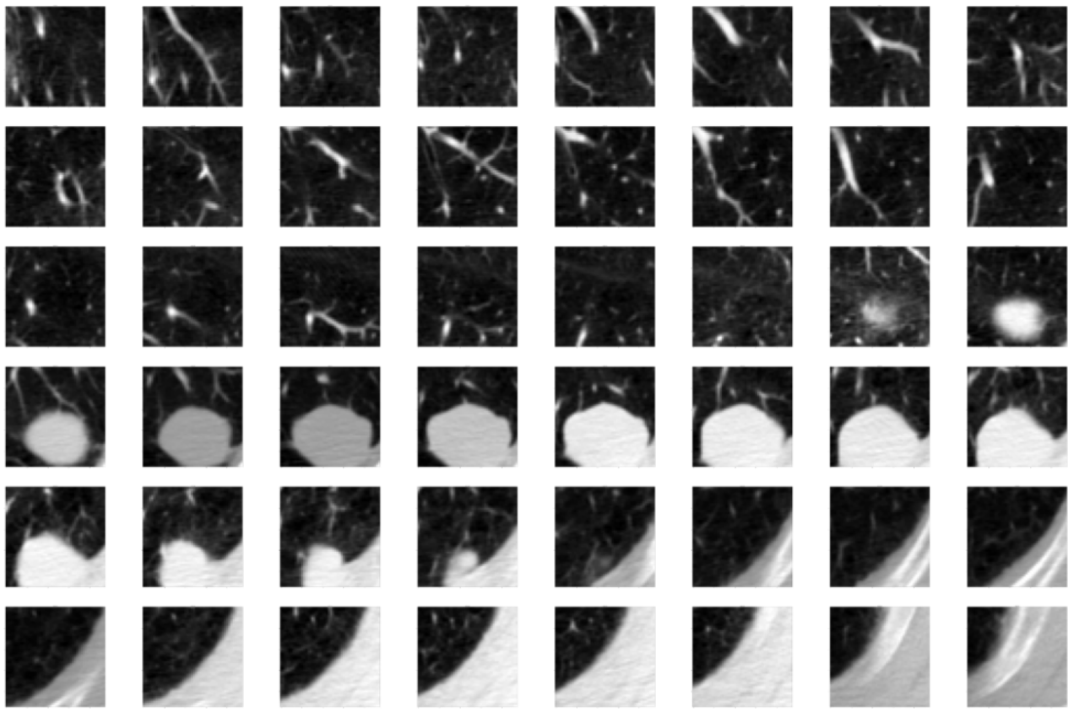
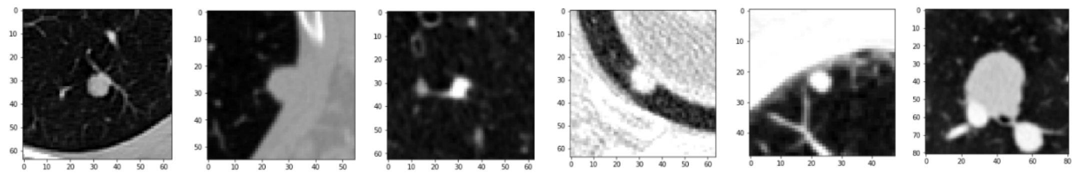


FIGURE 7.4: 2-D 64x64 pixels candidates of different nodules



than positives. For instance, candidates set provided by LUNA16 challenge consists of 549,714 non-nodules samples and only 1,351 nodules. The idea in this split is to give twice more negatives than positives for training and 10 times more non-nodules than nodules for testing, while keeping the number of positives candidates the same as in D1. This would show whether the network is capable to deal with data distribution which is closer to reality, in other words on the unbalanced set. More importantly, we will see if training on this split increases or decreases the model performance. We refer to this dataset division configuration as D2.

To conclude, for training data D2 contains 15,000 nodules candidates (same as in D1) and twice more non-nodules - 30,000. For testing 8,000 positives are used along with 80,000 negative samples. Selection process for non-nodules regions is identical to the one used in D1. This split consists of D2 testing set and D2 training set.

7.4 Training details

We tried three different batch sizes: 32, 64, 128, and empirically it was proven that 64 samples per batch is the best for training through all experiments. In some train-test split configurations we had more data for training than in others resulting in decrease of a batch size. For optimization in some experiments we used Adam and in others RMSprop with learning rate varying from 0.0001 to 0.0005 and momentum=0.95. During training we reduce learning rate by multiplying it by 0.2 when validation accuracy reaches plateau (doesn't improve through 5 iterations).

We also tried setting the class weights inside the loss for the training on an imbalanced dataset. However, it did not give major improvements, this is why we don't report the experiments which had this adjustment.

7.5 Results

We report measured performance of all the methods in Table 7.1. The sensitivity metric is the number of true positives divided by the sum of false positives and true positives. Test sets from D1 and D2 differ in the proportion of positive class to negative, so the results in the upper part of this table should not be directly compared with those in the lower part. This table, nonetheless, shows the performance of different networks. We can clearly observe the increase in accuracy and sensitivity after going from 2-D convolution to 3-D convolution. The results also correlate with the complexity and depth of the network, but the biggest influence lies onto the choice of the training data. The fact that we see lower sensitivity on the highly imbalanced data (10 to 1 proportion negatives to positives) is expected. We chose this split as it is more consistent with the real world, so we can see a better image of what the final score will look like at the 888 scans from LIDC dataset. However, real dataset is even more unbalanced. After training we observed that networks learned to classify nodules, but still make mistakes producing false positives.

To really measure which of the splits is better to use for training, we should evaluate them on one unified test set. Results on the whole LUNA16 data are reported in Table 7.2. It shows that giving two times more negatives than positives for training (D2 configuration) produces better performance than training on the balanced dataset (D1).

TABLE 7.1: The performance summary. Each network in this table was trained and tested on the data mentioned in column Dataset. How those sets were obtained is described in 7.3.

Model name	Dataset	Test accuracy (%)	Test sensitivity (%)
CNN T4	D1	85.7	85.2
CNN T5	D1	86.5	85.4
VGGNet-11	D1	88.2	88.6
LeNet 3-D	D1	92.5	86.0
CNN T5 3-D	D1	94.2	92.0
VGGNet-11 3-D	D1	97.4	95.8
CNN T4	D2	92.1	77.3
CNN T5	D2	91.9	77.2
VGGNet-11	D2	96.0	82.7
LeNet 3-D	D2	96.9	89.5
CNN T5 3-D	D2	97.1	91.4
VGGNet-11 3-D	D2	99.0	95.1

We want to compare achieved results with other studies. For those we pick three works which showed state-of-the-art performance in LUNA16 challenge, for instance, Dou et al. [12] finished top-1 in LUNA16 competition, Torres et al. [37] was third runner-up, and van Ginneken et al. [39] entered top-5. Their approaches: [12] - three different contextual 3D ConvNets architectures (Archi-a, Archi-b, Archi-c), [37] - a feed-forward neural network trained on a set of 13 features, including spatial, intensity, and shape features, [39] - OverFeat, trained for object detection in natural images. Another work which we used for comparison is Li et al. [22], whose model architecture we implemented, trained, and took as a baseline. It is referred in this thesis as CNN T4 and its slight alteration - CNN T5, more about them is in 5.1.

It is hard to compare performance of different researched works because most of them do not perform evaluation on the whole LUNA16/LIDC dataset but select for test only a certain number of scans. However, we chose for comparison the studies which have been tested on approximately the same quantity of data giving an opportunity to see how well our implemented methods operate.

It is also very time consumptionnaly to run prediction step on a full testing set from LIDC scans each time we perform an experiment, for example, CNNT5 evaluation on 888 scans takes around 5 hours. This was another reason why we needed to form D1 and D2 test sets.

Below we describe and explain evaluated scores achieved in our work.

7.5.1 CNN T4 & CNN T5

The performance of these two networks is very similar as it is expected due to their architectures' resemblance [22]. However, CNN T5, which has more features maps on the first convolution layer than CNN T4 performs slightly better on the test data from D1 described in Section 7.3, but a little worse on the D2. For evaluation scores comparison see Table 7.1. More important results are the results on the full LIDC dataset in Table 7.2. It is clear that we did not achieve good results on 888 scans from LIDC using CNN T5 trained on D1 split. We believe that more research is needed to find the correct split for the training data to get better scores with this network.

Changes of training loss and accuracy for CNN T5 are shown in Figure 7.6. During the training and testing accuracy on both: D1 and D2 grows constantly.

7.5.2 VGGNet

On the test set from D1, we observe that trained on D1 VGG-11 gives better accuracy than CNN T4 and CNN T5. Moreover, when comparing its sensitivity score it becomes clear that this network performs better than other 2-D CNNs used in our work. By applying VGGNet we achieved over 88.6% sensitivity on D1 test: 16,000 regions (balanced 8,000 positives, 8,000 negatives), while CNN T5 shows 85.4% of sensitivity on the same data. When testing on D2 (imbalanced test set) with 88,000 images in total - D2 split, VGG-11 obtains 82.7% sensitivity.

When evaluating this network on LIDC data we find a big difference in results predicted by same model which was trained on different data splits: for D2 we obtained 72.1% recall at 8 FPs/scan rate, but for D1 only 42.0% at 8 FPs/scan. This proves the need of more experiments about ways to choose data for training, especially if this is 2-D images that we deal with. Because in two dimensional case we loose some information when we select only one slice to represent a 3-D candidate. During prediction on 2-D data we also take the center, which might contain some artifacts (like blood vessels), those can cause a false positive prediction. If we take a 3-D image of the same region, because of its shape, form and length the vessel starts to look more distinct from the nodule.

7.5.3 LeNet 3-D

LeNet 3-D yields much better test accuracy on both training datasets than any 2-D network in this work, see Table 7.1. On the other hand, on D1 test set it fails to outperform VGGNet-11 in sensitivity showing only 86% while deeper network - VGG-11 gets above 88% of overall sensitivity on D1. This, nonetheless, does not happen with D2, where LeNet 3-D proves to perform better than any 2-D net achieving 89.5% sensitivity.

On LIDC testing this network shows better results when trained on imbalanced - D2 training dataset - 78.1% sensitivity at 4 FPs/scan and 83% at 8 FP rate, which are already good scores that can be compared to other studies. In Figure 7.5 the whole FROC curve for LeNet 3-D is shown. We can also observe, that training on balanced data gives worse results, however, it shows better performance than much deeper, but 2-D network VGG-11.

7.5.4 CNN T5 3-D

Out of CNN T5 and CNN T4 we chose the first network to be converted to 3-D because it produced slightly better results in experiments on 2-D data. From this network we received 94.2% accuracy and 92% sensitivity on the test data from D1. This model shows similar performance to LeNet 3-D in terms of accuracy, but reports a much higher recall score. Moreover, it is significantly better than 2-D version of this same architecture. CNN T5 3-D outperforms CNN T5 by more than 6% on sensitivity score and 8% on D2 testing accuracy. Time evolution of the training process is presented in Figure 7.6.

7.5.5 VGGNet 3-D

This network shows the best performance on D1 for both: accuracy and sensitivity, 97.4%, 95.8% correspondingly. When comparing these results to the ones obtained by VGGNet-11 on the same test set (88.2% accuracy, 88.6% recall), we observe a huge improvement. This means that application of a 3-D network rather than 2-D helped to achieve better results, same conclusion is derived from the performance of CNN T5 and CNN T5 3-D. Change of the dimensionality, however, is not the only factor of this method's overall success. From Table 7.1 it is clear that the deeper network receives better scores, while LeNet 3-D and CNN T5 3-D, although use 3-D convolution, produce worse results than VGGNet 3-D.

VGGNet-11 3D receives best results on LIDC database: trained on D1: 82.1% at 4 FPs/scan and 87.0% on 8 FPs/scan rate; trained on D2: 90.6% at 4 FPs/scan and 91.9% on 8 FPs/scan rate. In Figure 7.5 the whole FROC curve for this method is shown.

TABLE 7.2: The performance comparison of different studies on scans from LUNA16/LIDC database (888 scans in total). The upper part of the table shows the evaluation scores of the models which we used and implemented, while the lower part consists of results reported by other CAD systems in [32]. * the number of scans used by Torres is bigger than size of LUNA16/LIDC because besides from the LIDC/IDRI database they also took 50 scans from ANODE09 and 20 from ITALUNG-CT. **Contains whole LIDC database including scans with slice thickness greater than 2.5 mm (those are not present in LUNA16)

Method	Training Data	Cases (#scans)	Sensitivity 4 FPs/exam	Sensitivity 8 FPs/exam
CNN T5	D1	888	28.0	36.0
VGGNet-11	D1	888	31.4	42.0
	D2	888	63.4	72.1
LeNet 3-D	D1	500	70.6	76.7
	D2	888	78.1	83.0
VGGNet-11 3-D	D1	888	82.1	87.0
	D2	888	90.6	91.9
Dou et al.	-	888	90.7	92.2
Torres et al.	-	949*	-	80.0
van Ginneken et al.	-	865	76	-
Li et al.	-	1010**	87.1	-

FIGURE 7.5: FROC analysis results achieved. This evaluation used 888 scans from LUNA16/LIDC database. The metric scores are obtained by running evaluation script from LUNA16 challenge. In Table 7.2 the reported sensitivity at FPs/scan is the average FROC score, but on the graphs below we also plot its upper and lower bound. Each figure is labeled with the name of a corresponding method

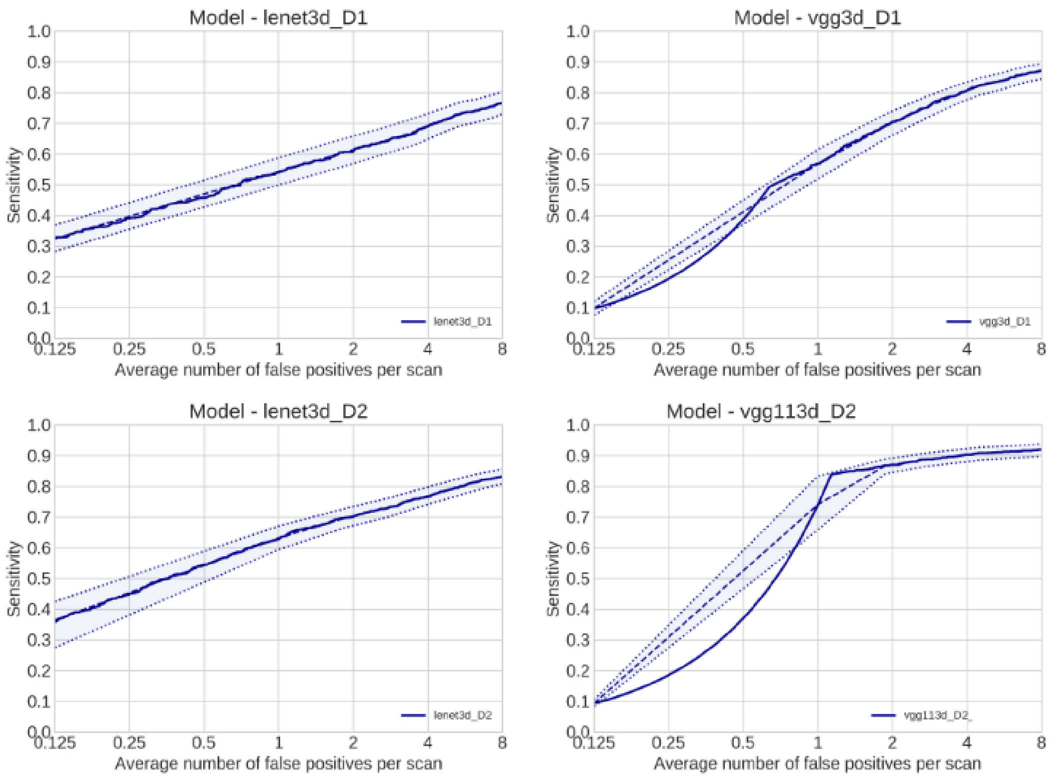
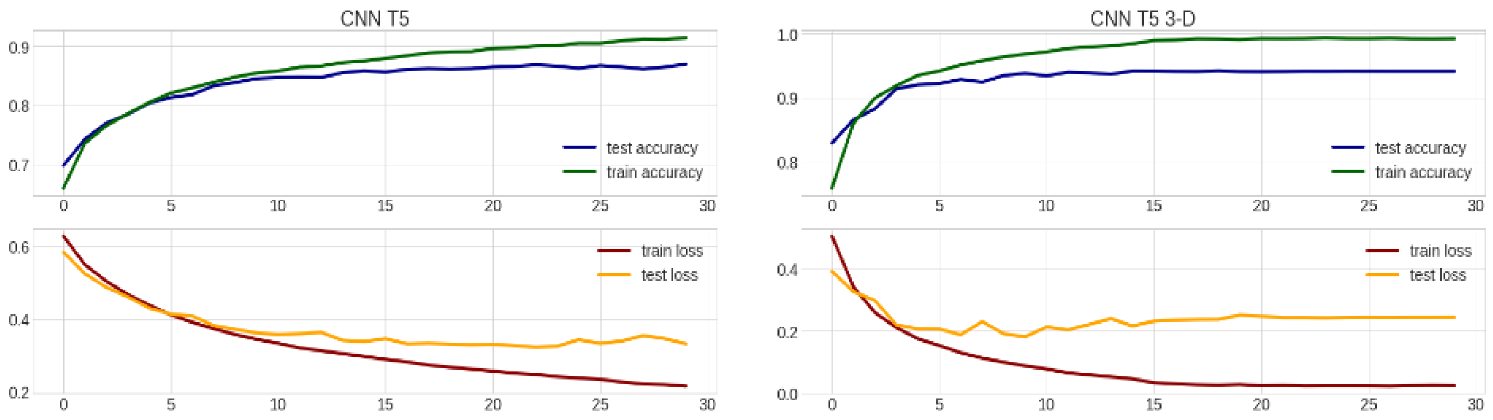


FIGURE 7.6: CNN T5 and CNN T5 3-D: training logs for 30 iterations (validation loss reaches plateau and stops decreasing after 30 epochs, other networks need more time to train), training and testing dataset configuration - D1 see 7.3 for details. This figure shows the training progress of two models which have only differ in the convolution dimensionality



Chapter 8

Conclusions

8.1 Results summary

From the obtained results we conclude that deeper 3-D convolutional network outperforms all of the other used methods. For instance, VGG-11 3-D showed significantly higher scores than CNN T5 3-D, LeNet 3-D, and each of their two dimensional versions. Its superiority was proven on the prepared test sets from D1 and D2, as well as on data from LIDC. Due to the evaluation on CT scans from LIDC, we could compare the performance across others studies, which were not replicated in this thesis, but reported their results in LUNA16 challenge [32]. Among all of our implemented methods 3-D networks show promising and competitive scores at both 4 FPs/scan and 8 FPs/scan rates. Unfortunately, 2-D networks failed to succeed. We believe that more time and research is required to find the best data configuration and model architecture to receive results using 2-D convolution as good as while using 3-D.

Nevertheless, we also conclude, how important it is to follow the proportion of classes in the training dataset. Networks trained on D2 split (twice more negatives samples than positives) proved to give better overall sensitivity score on LIDC data, than the ones trained on D1 (balanced set).

In this thesis we achieved all the goals which were set in the beginning of the project: explored and described previous works on nodule classification and detection, applied three models with 2-D convolution and three with 3-D, evaluated and tested their performance on different splits of the same dataset, and reported the comparable FROC sensitivity scores for the used models. Moreover, we implemented each neural network using two different frameworks separately. We also published the source code in a public repository, so it could be used for future work.

8.2 Future work

Possible ideas for future work:

1. Test the models' performance for different image sizes. In other words, prepare larger regions of interest, for example, 64x64x64 for 3-D and 64x64 correspondingly for 2-D.

2. Use 3-D convolutional neural networks which are deeper and more complex. Those include ResNet-3D and similar architectures.

3. Use deep neural networks to extract candidates regions from raw images. Such architectures as U-Net should be used.

4. Apply Multiple instance learning to use the entire scan as a bag of candidates and thus be less dependent from having very precise annotations.

8.2.1 Multiple instance learning

Not always it is possible to have annotations for each sample, even more, such labeling can take a lot of time and thus be expensive. Interpreting CT scans as bags and instances can help improve results of nodule detection. Some have already applied multi-instance learning to medical data: [2], [43], [14]. This is, however, not fully explored field and a promising area of research.

MIL was used for nodule detection by [2] in a following order: 1) images processing, 2) candidate ROI extraction, 3) adaptive bag construction, 4) feature extraction, 5) feature dimensional reduction by PCA, 6) classifier based on the proposed MIL-CAND. In this case two MIL algorithms: Diversity Density (DD) [25] and a Multiple Instance Learning with EM algorithm (EM-DD) [42] were applied.

We believe that applying Multiple instance learning combined with 3-D CNNs can improve the results of nodule detection task. This hypothesis should be tested in further research.

Bibliography

[1] Cancer Facts & Figures 2018. *American Cancer Society*, 2018.

[2] An et al. An improved framework for computer aided nodule detection based on multiple instance learning. *Chinese Conference on Pattern Recognition*, pages 413–421, 2012.

[3] Bergtholdt et al. Pulmonary nodule detection using a cascaded SVM classifier. *SPIE Medical Imaging*, 2016.

[4] F. Chollet et al. Keras. `https://keras.io`, 2015.

[5] Cohen and Welling. Group equivariant convolutional networks. *International Conference on Machine Learning (ICML)*, 2016.

[6] Ding et al. Accurate pulmonary nodule detection in computed tomography images using deep convolutional neural networks. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2017.

[7] Dolejsi et al. The lung TIME-annotated lung nodule dataset and nodule detection framework. *Proceedings of SPIE*, 7260:535–559, 2009.

[8] Dolejšı et al. Reducing false positive responses in lung nodule detector system by asymmetric Adaboost. *Proceedings / IEEE International Symposium on Biomedical Imaging: from nano to macro. IEEE International Symposium on Biomedical Imaging*, 2008.

[9] Dolejšı et al. The lung TIME—annotated lung nodule dataset and nodule detection framework. *Proceedings of SPIE - The International Society for Optical Engineering*, 2009.

[10] Dou et al. 3d deeply supervised network for automatic liver segmentation from ct volumes. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2016.

[11] Dou et al. Automatic detection of cerebral microbleeds from mr images via 3d convolutional neural networks. *IEEE Transactions on Medical Imaging*, 35 (5):1–1, 2016.

[12] Dou et al. Multi-level contextual 3D CNNs for false positive reduction in pulmonary nodule detection. *IEEE Transactions on Biomedical Engineering*, 99, 2016.

[13] I. A. for Research on Cancer Working Group. Tobacco smoke and involuntary smoking. *IARC Monographs on the Evaluation of Carcinogenic Risks to Humans*, 83:1–1438, 2004.

[14] Hussein et al. Risk stratification of lung nodules using 3D CNN-based multitask learning. *International Conference on Information Processing in Medical Imaging*, pages 249–260, 2017.

[15] Jacobs et al. Automatic detection of subsolid pulmonary nodules in thoracic computed tomography images. *Medical Image Analysis*, 18:374–384, 2014.

[16] Kamnitsas et al. Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. 2016.

[17] Karpathy. CS231n convolutional neural networks for visual recognition. *Stanford University*, 2016.

[18] Kauczor et al. ESR/ERS white paper on lung cancer screening. *European Respiratory Journal*, 25(9):2519–31, 2015.

[19] Korolev et al. Residual and plain convolutional neural networks for 3D brain MRI classification. *IEEE International Symposium on Biomedical Imaging 2017*, 2017.

[20] Kumar et al. Lung nodule classification using deep features in CT images. 2015.

[21] LeCun et al. Object recognition with gradient-based learning. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[22] Li et al. Pulmonary nodule classification with deep convolutional neural networks on computed tomography images. *Computational and Mathematical Methods in Medicine*, 2016:1–7, 2016.

[23] LIDC/IDRI. https://wiki.cancerimagingarchive.net/display/public/lidc-idri. 2016.

[24] LUNA16. https://luna16.grand-challenge.org/. 2016.

[25] Maron et al. A framework for multiple-instance learning. 1998.

[26] Murphy et al. A large scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbour classification. *Medical Image Analysis*, 13:757–770, 2009.

[27] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS-W*, 2017.

[28] Ren et al. Towards real-time object detection with region proposal networks. in advances in neural information processing systems. *NIPS*, 2015.

[29] Russakovsky et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[30] Setio et al. Automatic detection of large pulmonary solid nodules in thoracic CT images. *Medical Physics*, 42:5642–53, 2015.

[31] Setio et al. Pulmonary nodule detection in CT images: false positive reduction using multi-view convolutional networks. *IEEE Transactions on Medical Imaging*, 2016.

[32] Setio et al. Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: the luna16 challenge. *Medical image analysis*, 2016.

[33] Shi et al. A clinicopathological study of resected non-small cell lung cancers 2 cm or less in diameter: a prognostic assessment. *Med Oncol*, 28:1441–1446, 2011.

[34] Stewart and Wild. World cancer report 2014. *International Agency for Research on Cancer*, 2014.

[35] Tan et al. A novel computer-aided lung nodule detection system for CT images. *Medical Physics*, 38 (10):5630–45, 2011.

[36] Tan et al. Phased searching with neat in a time-scaled framework: Experiments on a computer-aided detection system for lung nodules. *Artificial Intelligence in Medicine*, 2013.

[37] Torres et al. Large scale validation of the M5L lung CAD on heterogeneous CT datasets. *Medical Physics*, 42 (4):1477, 2015.

[38] van Ginneken et al. Comparing and combining algorithms for computer-aided detection of pulmonary nodules in computed tomography scans: the ANODE09 study. *Medical Image Analysis*, 14:707–722, 2010.

[39] van Ginneken et al. Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans. 2015.

[40] Winkels and Cohen. 3D G-CNNs for pulmonary nodule detection. 2018.

[41] Yang et al. Visual explanations from deep 3D convolutional neural networks for Alzheimer's disease classification. 2018.

[42] Zhang et al. EM-DD: An improved Multiple-Instance Learning technique. 2002.

[43] Zhu et al. Deep multi-instance networks with sparse label assignment for whole mammogram classification. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 603–611, 2017.