University of Calgary

PRISM Repository **https://prism.ucalgary.ca**

The Vault Open Theses and Dissertations

2018-01-16

DeepCADe: A Deep Learning Architecture for the Detection of Lung Nodules in CT Scans

Golan, Rotem

Golan, R. (2018). DeepCADe: A Deep Learning Architecture for the Detection of Lung Nodules in CT Scans (Doctoral thesis, University of Calgary, Calgary, Canada). Retrieved from https://prism.ucalgary.ca. http://hdl.handle.net/1880/106312

Downloaded from PRISM Repository, University of Calgary

UNIVERSITY OF CALGARY

DeepCADe: A Deep Learning Architecture for the Detection of Lung Nodules in CT Scans

by

Rotem Golan

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN COMPUTER SCIENCE

CALGARY, ALBERTA

January, 2018

© Rotem Golan 2018

Abstract

Early detection of lung nodules in thoracic Computed Tomography (CT) scans is of great importance for the successful diagnosis and treatment of lung cancer. Due to improvements in screening technologies, and an increased demand for their use, radiologists are required to analyze an ever increasing amount of image data, which can affect the quality of their diagnoses. Computer-Aided Detection (CADe) systems are designed to assist radiologists in this endeavor.

In this thesis, we present DeepCADe, a novel CADe system for the detection of lung nodules in thoracic CT scans which produces improved results compared to the state-of-the-art in this field of research. CT scans are grayscale images, so the terms scans and images are used interchangeably in this work. DeepCADe was trained with the publicly available Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) database, which contains 1018 thoracic CT scans with nodules of different shape and size, and is built on a Deep Convolutional Neural Network (DCNN), which is trained using the backpropagation algorithm to extract volumetric features from the input data and detect lung nodules in sub-volumes of CT images.

Considering only lung nodules that have been annotated by at least three radiologists, DeepCADe achieves a 2.1% improvement in sensitivity (true positive rate) over the best result in the current published scientific literature, assuming an equal number of false positives (FPs) per scan. More specifically, it achieves a sensitivity of 89.6% with 4 FPs per scan, or a sensitivity of 92.8% with 10 FPs per scan. Furthermore, DeepCADe is validated on a larger number of lung nodules compared to other studies (Table 5.2). This increases the variation in the appearance of nodules and therefore makes their detection by a CADe system more challenging.

We study the application of Deep Convolutional Neural Networks (DCNNs) for the detection of lung nodules in thoracic CT scans. We explore some of the meta parameters that affect the performance of such models, which include:

1. the network architecture, i.e. its structure in terms of convolution layers, fully-connected layers, pooling layers, and activation functions,

2. the receptive field of the network, which defines the dimensions of its input, i.e. how much of the CT scan is processed by the network in a single forward pass,

3. a threshold value, which affects the sliding window algorithm with which the network is used to detect nodules in complete CT scans, and

4. the agreement level, which is used to interpret the independent nodule annotations of four experienced radiologists.

Finally, we visualize the shape and location of annotated lung nodules and compare them to the output of DeepCADe. This demonstrates the compactness and flexibility in shape of the nodule predictions made by our proposed CADe system. In addition to the 5-fold cross validation results presented in this thesis, these visual results support the applicability of our proposed CADe system in real-world medical practice.

Acknowledgements

We acknowledge the National Cancer Institute and the Foundation for the National Institutes of Health, and their critical role in the creation of the free publicly available LIDC-IDRI Database used in this study [AI+11; Cla+13; Tea15]. The LIDC-IDRI database allowed us to employ a machine learning approach to the detection of lung nodules in CT scans, and compare the performance of our CADe system to the work of others who also used the LIDC-IDRI database to evaluate the performance of their systems.

We thank Alberta Innovates Technology Futures (AITF), who funded this research as part of their Graduate Student Scholarships program. Without such support it would have been immensely more difficult for me to complete this thesis, and I greatly appreciate their support. We hope that this work will take CADe systems a step forward on their path to becoming integrated in the daily practice of radiologists, and consequently to improve the health care of patients. There is still a long way to go on this path, but we hope that this work will encourage the medical community to share more of their annotated medical imaging data since this has great potential in improving patient care.

We thank Zebra Medical Vision (https://www.zebra-med.com), who provided us with the computational resources necessary to run our experiments and precious advice in the early stages of this research. After about a year of collaboration we both went our separate ways, but we still appreciate their support and the interesting discussions we had with their associated radiologists and technical team.

Finally, we thank eXDee Ltd. (http://www.exdee.ai) for a great internship experience, which was supported by Mitacs (https://www.mitacs.ca)through the Mitacs Accelerate program, and their contribution to this research through providing us with the computational resources necessary to run our experiments during the final stages of this work.

Table of Contents

List of Tables

List of Figures and Illustrations

List of Algorithms

List of Symbols, Abbreviations and Nomenclature

Symbol	Definition
DL	Deep Learning
NN	Neural Network
DNN	Deep Neural Network
CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neural Network
FCN	Fully Convolutional Network
ML	Machine Learning
FML	Feature-based Machine Learning
PML	Pixel-based Machine Learning
DICOM	Digital Imaging and Communications in Medicine
CT	Computed Tomography
CADe	Computer-Aided Detection
DeepCADe	The name of our proposed CADe system
VOI	Volume of interest
MIP	Maximum Intensity Projection
SVM	Support Vector Machine
SOM	Self Organizing Map
FLD	Fisher Linear Discriminant
PCD	Persistent Contrastive Divergence Algorithm
ROC	Receiver Operating Characteristic Curve
FROC	Free-Response Receiver Operating Characteristic Curve
ReLU	Rectified Linear Unit
tanh	Hyperbolic Tangent

sig	Logistic Function
loss	Loss Function
h	Hypothesis Function
θ	Network Parameters
α	Learning Rate
λ	Regularization Parameter
TPR	True Positive Rate
FPR	False Positive Rate
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

Chapter 1

Introduction

One of the hallmarks of human cognition is our ability to recognize patterns within the constant stream of signals flowing into our nervous system through its various sensory organs. For example, light waves which come from the sun, fire, and other light sources around us hit our retina after bouncing back from the environment around us, and are being converted into electrical signals. These electrical signals are transmitted to the so called visual cortex of our brains with the help of numerous neurotransmitter molecules and through a cascade of neurons firing action potentials along this path. All this is happening in a fraction of a second and in an immediate response to visual stimuli which appear within our receptive fields.

In 1950, Allan Turing developed a test, known as the Turing test, which would determine whether a machine can exhibit intelligent behavior equivalent to, or indistinguishable from, that of a human [Tur50]. In its simplest form, there are three participants: a human evaluator, a human, and a machine which is designed to generate human-like responses as part of a text based natural language conversation. The goal of the test is to examine whether the human evaluator can distinguish between the human and the machine based on their conversations. While this test is about testing a machine's ability to conduct a natural language conversation, the underlining question is the same as to whether a machine can "see", or in other words, whether it can recognize the objects that are found in its receptive field the same way a human can.

Making computers "see" is the holy grail of the field of Computer Vision which is one of the cornerstones of Artificial Intelligence. When a picture of a cat is input to a machine, the machine does not actually "know" there is a cat in the picture. All it has is a matrix

of pixel values. So for example, if the input image is in grayscale, there is a single matrix with intensity values in the range 0 to 255, or if it is a RGB image, it is represented as three matrices (3 channels), each containing the intensity values for a specific color, namely red, green, and blue. One of the goals of Computer Vision is to process this raw image data in order to detect, segment, manipulate, and recognize the objects it contains.

Traditionally, image processing systems would rely on hand-engineered features to recognize objects in an image. There is no exact definition of what constitutes a feature, since different kinds of images contain different kinds of objects, but in general a feature can be defined as an "interesting" part of an image. Examples for this vary and include edges of various shapes, corners, blobs, and regions of interest. Since feature extraction is, in most cases, used as a starting point for computer vision algorithms, and since the performance of these algorithms heavily depends on the quality of features extracted, finding the best set of features for solving a specific vision task is of utmost importance. Consequently, much research has been done in that area, and a very large number of feature detectors have been developed throughout the years [TM+08].

Developing hand-engineered feature detectors has shown to be successful in many object recognition tasks, but it does have three major drawbacks: (1) its dependence on human experts, which is time consuming and expensive, (2) the challenge of choosing the ideal features that are needed to solve a specific recognition task, and (3) the challenge of coming up with higher-level features that would be extracted on top of lower-level ones. Machine Learning (ML) and Deep Learning are active research fields which offer some remedies to these drawbacks. For example, in both fields of research, the recognition of patterns and regularities in data is performed by means of iteratively "learning from examples". Also, in the field of Deep Learning, high-level abstractions in data are captured by using multiple processing layers composed of multiple linear and non-linear transformations.

Many deep learning systems have been developed in recent years and have indeed shown tremendous results in many computer vision tasks [KSH12; Sze+14; CMS12; NHH15; Che+14a; LSD15]. This recent trend did not come about from the development of better computational models, even though there have been considerable advancements in this area of research. Rather, it came about from two other major contributing factors, namely (1) the sharp increase in computational power capabilities such as CPU and GPU technologies, and (2) the explosion of labeled data, which have come with the rise of the Internet and the massive adoption of smartphone and other digital devices by the majority of the world's population.

While some deep learning systems have indeed eliminated the need for experts to develop hand-engineered feature detectors, they still rely on large-scale labeled data and therefore, still rely on human knowledge to annotate this data. However, this has not been an obstacle in many vision and other cognition tasks since it is easier to, for example, determine whether a particular image contains a cat, compared to determining what are the features, in all abstraction levels, that are needed to decide whether the image contains a cat. This is particularly true when you have billions of people constantly commenting and sharing images and other forms of media throughout the internet, essentially providing the research community with a reservoir of knowledge that need to be incorporated into decision making systems. This, in essence, has been the major contribution that deep learning has made in recent years, namely the utilization of massive amounts of data that are out there and incorporating them into machine learning systems.

While labeled image data is abundant in databases such as ImageNet [Den+09; Rus+14], which currently contains more than 14 million labeled natural images with more than 21 thousand sets of synonym objects, labeled data in other disciplines is scarce. Annotated chest Computed Tomography (CT) scans, for example, are hard to come by. Even though thousands of radiologists are waking up every morning to do their job, examining many thoracic CT scans, and segmenting, at least in their minds, various abnormal masses, this

enormous body of knowledge is not becoming accessible to the academic research community and the public. The reasons are numerous, but it starts with the very important requirement to protect patient private information, and continues with the conservative nature of the medical establishment to share data and adopt new technologies.

Nevertheless, there are a number of initiatives to change this state of affairs. One of them is the Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) database, which is a publicly available reference for the medical imaging research community [AI+11; Cla+13; Tea15]. Its aim is to support the development of computer-aided diagnostic methods for lung nodule detection and classification, and is the result of a collaboration between seven academic centers and eight medical imaging companies. While much effort has been put to this initiative, it only contains 1018 thoracic CT images that originated from a total of 1010 patients. We utilize this database in our work in order to examine whether this amount of well-annotated data is enough to bring about the fruits of deep learning to the task of detecting lung nodules in thoracic CT scans.

1.1 Thesis Overview

This thesis is organized as follows. Chapter 1 provides an introduction to the detection of lung nodules in CT scans and describes in detail the LIDC-IDRI database which our machine learning system is based on. Chapter 2 discusses a number of machine learning architectures and algorithms which are necessary for the understanding of later chapters. Chapter 3 provides a survey and critical assessment of related work done by other research groups, and examines how it is related to this work. Chapter 4 describes the deep learning architectures we examined to detect lung nodules in CT scans, and how they are used to compute a three dimensional voting grid, with which the location and boundary of lung nodules are predicted. Chapter 5 describes the experimental results of our work. It includes a systematic study on the use of DCNNs for the lung nodule detection task. A Free-response

Receiver Operating Characteristic (FROC) analysis is used for performance evaluation in relation to previous work where the LIDC-IDRI dataset has also been used for validation. Finally, Chapter 6 concludes our work and discusses some future extensions of it.

1.2 The Detection of Lung Nodules in CT Scans

According to the American Cancer Society, lung cancer is the leading cause of cancer related deaths in the United States [SMJ15]. It was estimated that lung and bronchus cancers alone will cause 158,040 deaths in the United States in the year 2015, which is slightly more than the total number of deaths caused by brain/nervous system, breast, prostate, colon/rectum, and liver cancers combined. In addition, the 5-year survival rate of lung cancer is one of the lowest compared to other types of cancer, and is estimated to be 18% for years 2004 to 2010. Individual prognosis heavily depends on the extent of the disease at the time of diagnosis. So for example, if the tumor is detected while it is still small and localized, then the 5-year survival rate is 54%; but if it is detected at a later stage, when metastases have already developed and the tumor becomes either regional or distant, then the survival rate drops to 27% and 4%, respectively.

Unfortunately, most diagnoses occur at later stages of the disease, mainly due to lack of symptoms in its early stages. This has raised the idea of instituting widespread lung cancer screening as a matter of public health policy, which has been examined by a number of research institutions. One of these institutions is the U.S. National Cancer Institute (NCI), which sponsored the National Lung Screening Trial (NLST) and concluded that there is a statistically significant 20.3% relative reduction in lung cancer mortality when using low-dose helical computed tomography (LDCT) scans as a screening modality compared to using chest x-ray [Kra+11]. Therefore, it has been suggested to make LDCT, and CT in general, the preferred screening modality for early detection and diagnosis of lung cancer.

However, because of concerns about the radiation harms associated with CT scans and other medical imaging tests [Gon+09; Faz+09; SSB10; SB+10; SB+09], lower doses of radiation are often used in the screening setting despite higher resolution achievable with increased doses. In most cases, cancer takes many years to grow, and requires an environment that is supportive of its development [CCI16]. Factors such as diet, exposure to toxins and radiation, exercise, and mental health have all shown to have an effect on the progression or regression of cancer. Consequently, isolating and studying the effects of a single factor such as radiation exposure from medical imaging is challenging, and arguments about the risks associated with any potential carcinogen should be examined carefully.

A thoracic CT scan combines a series of X-ray images taken from different angles, and uses computer processing to create cross-sectional images, or slices, of the bones, blood vessels and soft tissues inside the chest. This results in a 3 dimensional image of the chest, where each volumetric pixel (voxel) has an attenuation value that is indicative to the type of material (tissue) present in its location.

As the resolution of CT screening technologies increases, and their demand, especially in the developing world, is on the rise, radiologists are overwhelmed with the amount of data they are required to analyze [McD+15]. This has the potential to cause fatigue among radiologists, and therefore affect the quality of their diagnoses. Computer-Aided Detection (CADe) systems have been developed in recent years to assist radiologists with this challenge [GJD16; Ric+11; Gol+09; MHR10; Tan+11; Ani+16]. Their goal is to provide radiologists with a second opinion, and support them in their interpretation of medical images. For example, a successful CADe system might detect lung nodules which would otherwise be overlooked by the radiologist, and bring these to the attention of the radiologist for further examination. However, if a CADe system produces too many false positives[1] (FPs), the radiologist's trust in the system can be undermined.

[1]The false positives of a detection system are the nodule predictions it makes, which do not overlap with any nodule annotation.

In a comprehensive survey conducted by Suzuki et al. [Suz13], three classes of classification techniques, into which the various CADe systems can be categorized, were identified. These are (1) Feature-based Machine Learning (FML), (2) Pixel/voxel-based Machine Learning (PML), and (3) Non-ML-based methods. Our CADe system [GJD16] belongs to the PML class of classification techniques since its volumetric features are trained in a supervised manner from the input data (i.e. voxel values of CT images). This is different from other CADe systems [Ric+11; Gol+09; MHR10; Tan+11], which belong to the FML class of classification techniques since their features are predetermined and are set manually by the designers of the system. Non-ML-based methods are defined as methods that do not use ML techniques. This includes all methods that do not have a "learning from examples" component in them.

The detection of lung nodules in CT scans is no easy task. One has to overcome the significant variability in the input data when approaching this task. First, since CT scanners are manufactured by different companies and are deployed with a wide range of radiation doses, they can vary in the way image reconstruction is performed and in the amount of image noise being generated [Mai+15]. Furthermore, CT scanners can be deployed with different configurations, so the images they produce can have values, such as slice thickness, pixel spacing, and image orientation, that are different from one another. Second, the size and shape of normal anatomical structures in the scans varies among different patients, and the CT images might contain artificial artifacts such as pace makers and artificial valves. Finally, lung nodules can vary in their appearance, from round solid objects to flat and liquid-like objects.

Lung nodules are small masses of tissue in the lung. They are usually about 5 millimeters to 30 millimeters in size. A larger lung nodule, such as one that is 30 millimeters or larger, is more likely to be cancerous than is a smaller lung nodule. Figure 1.1 shows an example of a CT image in which two lung nodules are present. The nodule boundaries are marked in red.

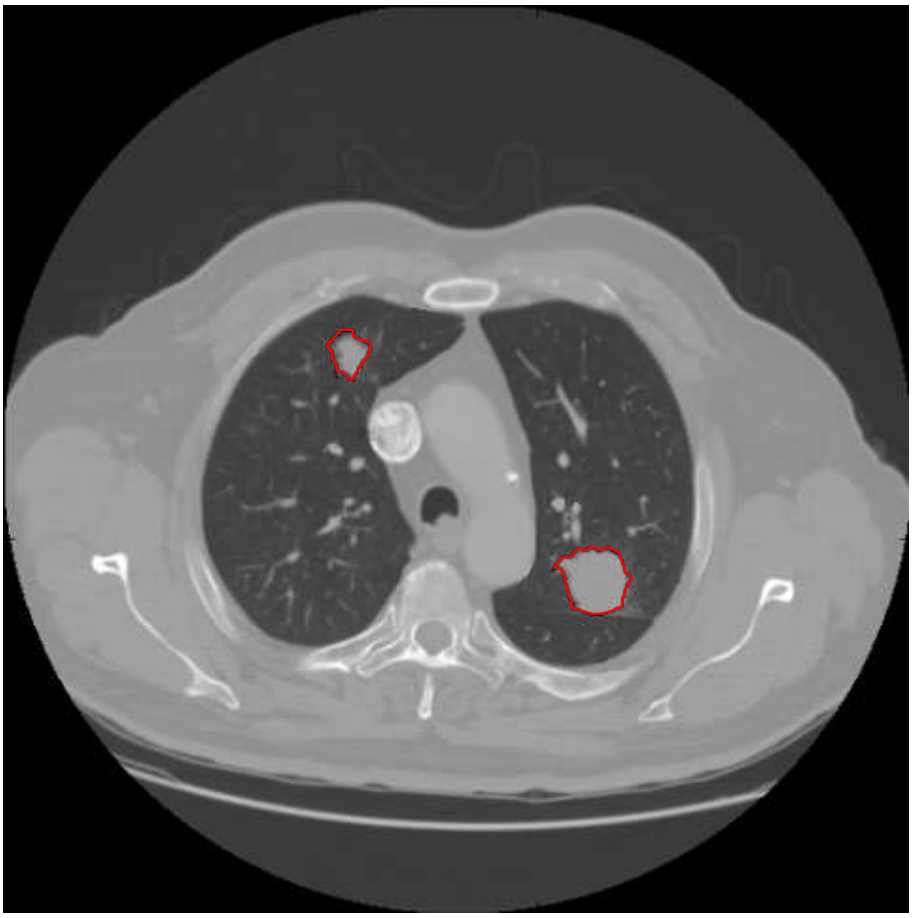

Figure 1.1: An example of a CT image which contains two lung nodules. The nodule boundaries are marked in red.

Modeling objects in CT images, such as lesions and organs, based on a simple model with a relatively small number of parameters, is unlikely to be sufficient to represent their complex structures. This means that Non-ML-based classifiers are probably not the right approach when tackling this task. However, FML and PML techniques have the potential of producing more complex models based on training examples, and therefore have a better chance of overcoming this challenge of variability.

In this thesis, we present a novel CADe system, DeepCADe, for the detection of lung nodules in thoracic CT scans which shows improved results over the state-of-the-art in this application domain. Our system is trained with images from the publicly available Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) database,

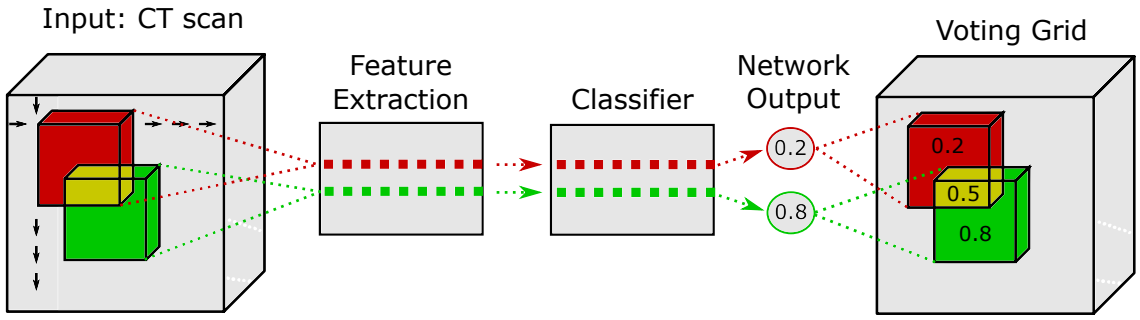

Figure 1.2: A high-level diagram of our CADe system. The red and green sub-volumes represent two overlapping receptive fields of the network.

which contains 1018 thoracic CT scans of individuals at different stages of their disease.

We use a Deep Convolutional Neural Network (DCNN), which is trained, using the backpropagation algorithm [RHW88], to detect lung nodules in sub-volumes of CT images. The DCNN is composed of two modules, namely *feature extraction* and *classification*. The first module is designed to extract valuable volumetric features from the input data, and the second one is the classifier which is expected to perform the high-level reasoning of the neural network. Once training is done, a sliding window algorithm is used to perform the detection on a complete CT scan. Figure 1.2 illustrates a high-level diagram of our CADe system. It illustrates how a single network is applied to two overlapping receptive fields (sub-volumes of a CT scan), and how the output of the network is aggregated into a single voting grid which describes the predicted probability of a nodule in each location in the original scan.

If we consider only those test nodules that have been annotated by at least three radiologists [2], our CADe system achieves a 2.1% improvement in sensitivity (true positive rate) over the best result in the current published scientific literature, assuming an equal number of false positives (FPs) per scan. More specifically, our CADe system achieves a sensitivity of 89.6% with 4 FPs per scan, or a sensitivity of 92.8% with 10 FPs per scan.

Furthermore, our CADe system is validated on a larger number of lung nodules compared to other studies. This increases the variation in the appearance of nodules and therefore

[2]The LIDC-IDRI dataset contains independent nodule annotations of four experienced radiologists. Section 1.3 gives a more detailed description of the dataset.

makes their detection by a CADe system more challenging. More specifically, our CADe system is validated on 279 annotated lung nodules, while other studies are validated on between 38 and 143 annotated lung nodules.

Finally, we perform a systematic study on the application of Deep Convolutional Neural Networks (DCNNs) for the detection of lung nodules in thoracic CT scans. We explore some of the meta parameters that affect the performance of such models, which include:

1. The depth of the classifier module of the DCNN, i.e. the number of fully-connected layers the classifier is composed of.

2. The size of the classifier module of the DCNN, i.e. the number of learnable parameters the classifier is composed of.

3. The depth and size of the feature extraction module of the DCNN, i.e. the effect convolutional layers have on the performance of the CADe system.

4. The benefit of using a rectified linear unit (ReLU) activation function compared to a sigmoid and hyperbolic tangent (tanh) functions.

5. The receptive field of the network, which defines the dimensions of its input, i.e. how much of the CT scan is processed by the network in a single forward pass.

6. Two ways to control the nodule prediction size, i.e. by using a dedicated threshold value or ignoring nodule predictions that are larger than a predefined size.

7. The four agreement levels, which define four sets of nodule annotations according to the level of agreement among four experienced radiologists.

1.3 The LIDC-IDRI Dataset

The Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) database is a publicly available reference for the medical imaging research community [AI+11; Cla+13; Tea15]. Its aim is to support the development of computer-aided diagnostic methods for lung nodule detection and classification, and is the result of a collaboration between seven academic centers and eight medical imaging companies.

The LIDC-IDRI dataset contains 1018 thoracic CT images that originated from a total of 1010 patients. The images comply with the Digital Imaging and Communications in Medicine (DICOM) standard. Images have a resolution of $[65, 764]\times512\times512$ voxels per scan, where $[65, 764]$ is the range of values for the number of slices in the 3D images. 512×512 is the in-plane pixel resolution of each of the 2D slices. The average number of slices per scan in the dataset is 240. The range of values for the slice thickness parameter is $[0.45, 5]$ mm with an average slice thickness of 1.73 mm. The range of values for the pixel spacing in each of the 2D slices is $[0.46, 0.97]$ mm with an average of 0.68 mm. These values help us understand the relationship between the image space (measured in voxels) and the real-world space (measured in mm). Figure 1.3 illustrates a $240\times512\times512$ CT scan which, assuming a slice thickness of 1.73 mm and pixel spacing of 0.68 mm, corresponds to a real-world sub-volume of size $41.5\times34.8\times34.8$ cm.

Each scan has been examined by four experienced thoracic radiologists in a two-phase image annotation process. In the initial blinded-read phase, each radiologist was asked to independently review the images in the dataset, and mark lesions they identified as (1) nodules ≥ 3 mm, (2) nodules < 3 mm, and (3) non-nodules ≥ 3 mm. In the second subsequent unblinded-read phase, each radiologist was asked to review their own marks, along with the anonymized marks of the three other radiologists, and make a final decision. The annotation files that are published as part of the LIDC-IDRI dataset contain only those annotations from the second phase of the annotation process.

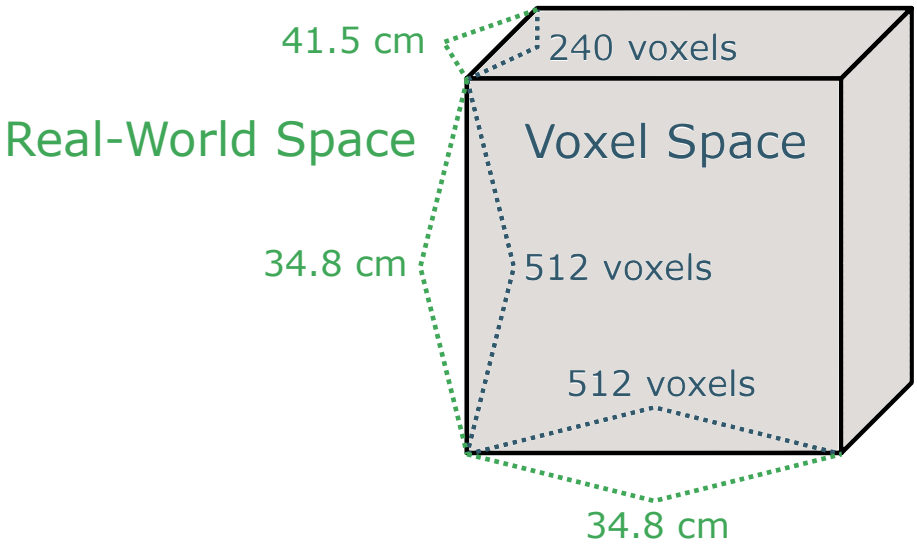

Figure 1.3: A 240×512×512 CT scan which, assuming a slice thickness of 1.73 mm and pixel spacing of 0.68 mm, corresponds to a real-world sub-volume of size 41.5×34.8×34.8 cm.

We chose to only use the annotations for the first category of nodules, namely nodules $\geq$ 3 mm. This set of nodules includes all those nodules in the LIDC-IDRI dataset with greatest in-plane dimension in the range $[3, 30]$ mm regardless of presumed histology. We decided to focus on this set of nodules since (1) all the papers we compare our results to have also focused on nodules $\geq$ 3 mm, (2) nodules that are greater or equal to 3 mm have a higher chance of being cancerous which makes them more clinically relevant, and (3) these nodules have their complete contour annotated, which we use to achieve a tighter detection box around the nodules. The other two sets of nodules, namely nodules $<$ 3 mm and non-nodules $\geq$ 3 mm, have only their center point annotated.

The goal of this two-phase annotation process was to identify as many lung nodules in each CT scan without forcing a consensus, i.e. without requiring the four radiologists to agree with each other and reach a unanimous decision about their annotations. But since no consensus was forced and the annotations are anonymized, one has to implement a grouping procedure, which determines which nodule annotations represent the same lung nodule and which are not. Our grouping procedure is described in Chapter 4.1. Once the grouping of nodule annotations is complete, we can associate each nodule with one of four agreement

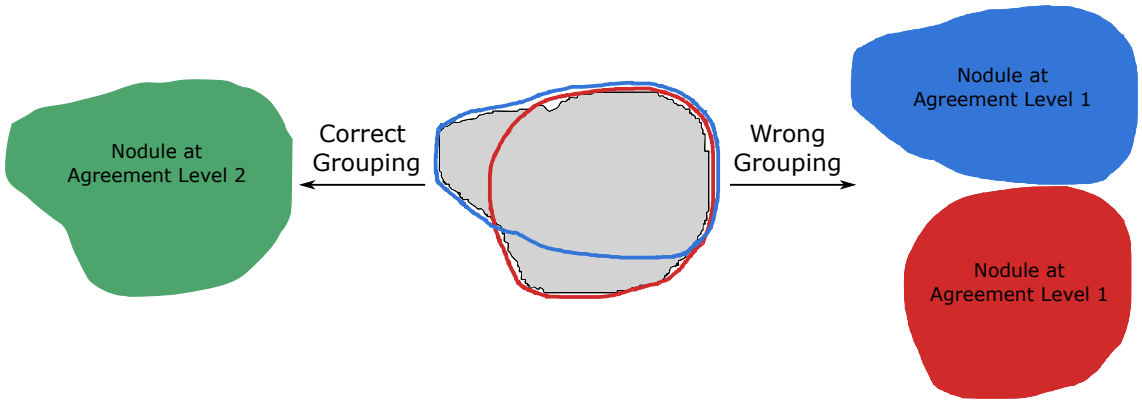

Figure 1.4: Two scenarios of grouping the independent annotations of two radiologists. The gray area which is bounded by a black contour represents the real nodule, and the blue and red contours represent annotations of two radiologists. The correct grouping leads to a single nodule at agreement level 2, while the wrong grouping leads to two separate nodules at agreement level 1. The two separate nodules overlap with each other spatially but are depicted as non-overlapping to emphasis the wrong interpretation of them as separate nodules at agreement level 1.

levels. Agreement level j, where $1 \leq j \leq 4$, includes all those nodules which were annotated by at least j radiologists.

Figure 1.4 illustrates the need for grouping the independent nodule annotations and the concept of agreement levels. It shows two scenarios of grouping the independent annotations of two radiologists, which we assume were aiming to mark the boundaries of the same nodule. This is not specified in the dataset, so one has to determine whether these independent annotations represent the same nodule or not using a grouping procedure. In one scenario, a correct grouping procedure considers both annotations to represent the same nodule. In the other scenario, a wrong grouping procedure considers both annotations to represent two separate nodules. Consequently, the correct grouping procedure will consider the union of the two annotations to be a single merged nodule annotation at agreement level 2. In contrast, the wrong grouping procedure will consider each annotation to be independent, which leads to two separate annotations, each of which is at agreement level 1.

A wrong grouping procedure will lead to an incorrect validation of any CADe system. For example, the wrong grouping procedure mentioned above leads to two nodule annotations

at agreement level 1. Therefore, if a CADe system makes a correct nodule prediction which "hit" both annotations, then the number of true positives will falsely increase by 2 instead of 1. If no nodule prediction is made to "hit" these two annotations, then the number of false negatives will falsely increase by 2 instead of 1.

Chapter 2

Background

Machine learning algorithms have been used to perform a variety of tasks such as supervised and unsupervised learning tasks [Ben09; BCV13]. In supervised learning, the goal is to learn a function from labeled training data. This means that each example in the training data is a pair consisting of an input object and a desired output value. In contrast, the goal in unsupervised learning tasks is to learn a function which provides a good internal representation of the input. Hence, the training data consists of input objects only without any desired output values. In this thesis, we examine the supervised learning task of classifying sub-volumes of CT scans into two classes, either containing a nodule or not. Then we use the resulting function to detect lung nodules in complete CT scans as illustrated in Figure 1.2.

In this chapter, we give a general definition of what a classification task is, and then continue by exploring a number of ways for approaching it. We describe two machine learning algorithms, namely logistic regression and convolutional neural networks (CNNs). Then, we describe the backpropagation algorithm [RHW88], which is a supervised learning algorithm, and see how it is used to learn (train) the weights (parameters) of deep neural networks (DNNs). DCNNs are one class of DNNs, and are considered "deep" since they are composed of multiple layers of non-linear transformations. Finally, we discuss the weaknesses of backpropagation and examine how recent developments in software and hardware technologies, and a significant increase in the size of available datasets, have allowed the machine learning community to overcome such weaknesses.

The notation used here is based on the online course in machine learning [Ng14] given by Andrew Ng. Throughout this chapter we use the words unit, node, and neurons interchange-

ably. Similarly, we use the terms weights and parameters of the network interchangeably. These are the building blocks of neural networks.

2.1 Classification

Classification is the problem of identifying to which of a set of classes a new input object belongs, based on a training set of data containing pairs of input objects and their respective class membership. More formally, given m training examples

$$(x^1 = (x_1^1, \ldots, x_n^1), c^1)$$

$$\vdots$$

$$(x^m = (x_1^m, \ldots, x_n^m), c^m)$$

where

$$(x_i^j \in X_i) \text{ and } (c^j \in C)$$

can we "learn" a hypothesis function

$$h : X_1 \times ... \times X_n \to C$$

which approximates the unknown relationship between the input objects and their respective class membership.

Ideally, the resulting hypothesis function h will generalize to examples that are outside of the training set. This means that given an input object $x = (x_1, \ldots, x_n)$ which is not in the training set, and given the class c it belongs to, h will return the correct mapping $h(x) = c$

2.2 Logistic Regression

We are given m training examples and their correct labels, denoted as $D = \{(x^1, c^1), ..., (x^m, c^m)\}$, where x^i is an $n + 1$ dimensional vector representing n features and a bias unit $x_0^i = 1$, and c^i is the output value of the i'th training example.

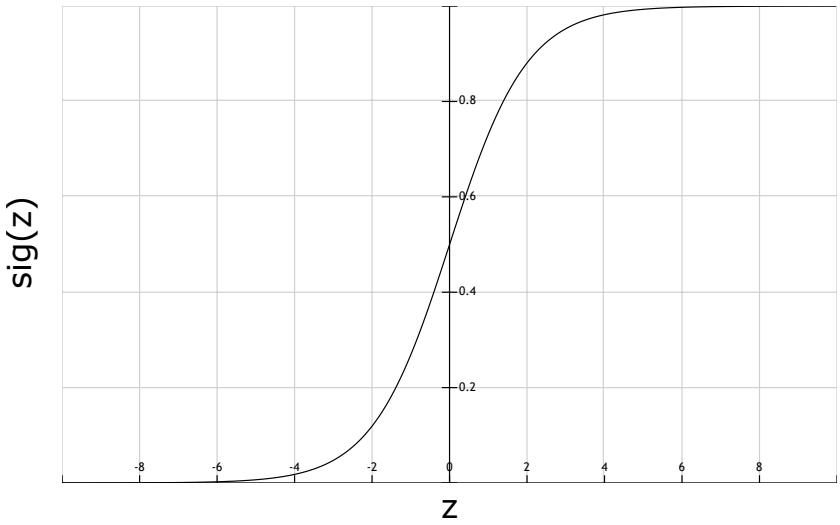

Figure 2.1: The logistic function $sig(z) = \frac{1}{1+e^{-z}}$

Logistic regression is one of the most basic supervised classification algorithms. Algorithm 1 describes the pseudo code of the gradient descent algorithm, which can be used to train a logistic regression model by defining the following three functions: a hypothesis function $h_\theta(x)$, a loss function $loss(\theta, D)$, and the derivative of the loss function $\frac{\delta}{\delta\theta_j} loss(\theta, D)$, where $\theta = (\theta_0, \theta_1, ..., \theta_n)$ is the parameter vector of the model. Assuming a binary classification task, we would like the hypothesis to be in the range $0 \leq h_\theta(x) \leq 1$. To do so, we use the logistic function $sig(z) = \frac{1}{1+e^{-z}}$, which returns values in the range $[0, 1]$, to define the following hypothesis:

$$h_\theta(x) = sig(\theta^T x). \tag{2.1}$$

Figure 2.1 plots the logistic function $sig(z)$. Next, we define a loss function, the sigmoid cross entropy loss function, as follows:

$$loss(\theta, D = \left\{(x^1, c^1), ..., (x^m, c^m)\right\}) = -\frac{1}{m}\sum_{i=1}^{m}(c^i \log h_\theta(x^i) + (1 - c^i)\log(1 - h_\theta(x^i))). \tag{2.2}$$

Figure 2.2 plots the loss function for a single example ($m = 1$, $D = \{(x, c)\}$) when (a) $c = 0$ and (b) $c = 1$. It shows that the loss function converges to zero as the hypothesis

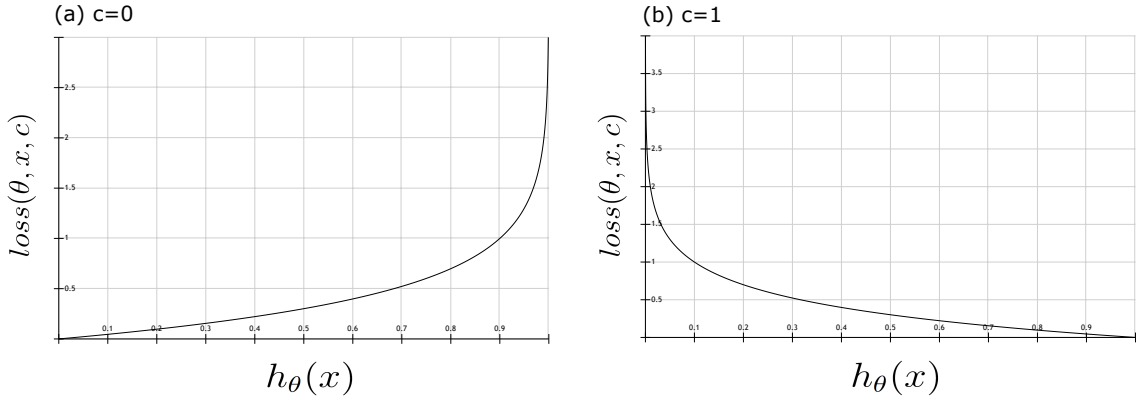

Figure 2.2: The loss function $loss(\theta, D = \{(x, c)\}) = -(c \log h_\theta(x) + (1 - c) \log(1 - h_\theta(x)))$ for a single example (x,c) when (a) $c = 0$ and (b) $c = 1$.

function $h_\theta(x)$ approaches the correct label c.

Finally, the derivative of the loss function is defined as follows:

$$\frac{\delta}{\delta\theta_j} loss(\theta, D = \left\{(x^1, c^1), ..., (x^m, c^m)\right\}) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - c^i)^2 x_j^i. \tag{2.3}$$

It is important to note that:

1. Given θ, the hypothesis function $h_\theta(x) = sig(\theta^T x)$ can be used to predict the label of new examples.

2. The bias unit allows for the shifting of the activation function, which may be critical for successful learning. Consequently, the activation function does not depend solely on its feature vector. The use of bias units is related to the batch normalization algorithm [IS15] which allows for both the shifting and scaling of the activation functions in neural networks. Similarly to the bias unit, the shift and scale parameters are learned as part of the learning algorithm.

3. As shown in Algorithm 1, α is the learning rate. If it is too small, $loss(\theta)$ will converge slowly, and if it is too high, $loss(\theta)$ might not converge at all.

4. If $loss(\theta)$ is a convex function, and assuming a reasonable α, the algorithm will always find an optimal solution with respect to $loss(\theta)$.

5. It is possible to get non-linear decision boundaries by adding higher order polynomials as new features.

6. It is possible to add a regularization component $\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$ to the loss function, where λ is the regularization parameter. High values of λ correspond to models with higher bias, i.e. error from incorrect assumptions in the learning algorithm, and low values of λ correspond to models with higher variation, i.e. error from sensitivity to small fluctuations in the training set.

Algorithm 1 Gradient Descent

1: Randomly initialize $\theta = (\theta_0, \theta_1, ..., \theta_n)$
2: Compute the hypothesis $h_\theta(x^i)$ for every $i = 1, ..., m$
3: Compute the loss function $loss(\theta, D = \{(x^1, c^1), ..., (x^m, c^m)\})$
4: **while** $loss(\theta, D)$ is decreasing **do**
5: Simultaneously update $\theta_j := \theta_j - \alpha\frac{\delta}{\delta\theta_j}loss(\theta, D)$ for every $j = 0, 1, 2, ..., n$
6: Compute the hypothesis $h_\theta(x^i)$ for every $i = 1, ..., m$
7: Compute the loss function $loss(\theta, D)$
8: **end while**

2.3 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a type of feed-forward neural network where the individual neurons are tiled in such a way that they respond to overlapping regions in its receptive field. It is inspired by models of the biological visual system, proposed by [HW62], and continues to be consistent with the modern understanding of the physiology of the visual system [Ser+07]. The first computational models based on these local connectivities between neurons are found in Fukushima's Neocognitron [Fuk80]. Fukushima found that when neurons with the same parameters are applied on overlapping regions of the previous layer, at different locations, a form of translational invariance is obtained. This allows CNNs to detect objects in their receptive field in a way that is invariant to their size, location, orientation, and other visual properties. In addition, this limited connectivity

of CNNs reduces the computational requirements necessary for their training compared to fully-connected neural networks.

On the other hand, the limited connectivity of CNNs implies that their input must be structured in such a way that weight sharing is appropriate. More specifically, the input values must not be independent of each other but rather they must be structured according to some temporal, spatial, or other kind of relationship. Examples of this include the spatial relationship of pixels in images, the temporal relationship of musical notes in audio tracks, or the spatial and temporal relationships of pixels in video tracks.

Figure 2.3 illustrates this limited connectivity of CNNs and how it differs from the connectivity in fully-connected neural networks. The fully-connected neural network that is to the left of the receptive field is composed of 36 learnable parameters and outputs 4 values. In contrast, the CNN to the right of the receptive field also outputs 4 values, but is composed of 4 learnable parameters having the shape of a 2x2 convolution kernel, which is applied on four overlapping regions of the receptive field.

CNNs were first trained using the backpropagation algorithm in [LeC+89], and ever since they have obtained state-of-the-art performance on several pattern recognition tasks. For example, large-scale CNNs were used to recognize objects in natural images as part of the ImageNet challenge [KSH12; Sze+14; Rus+14], for which they have shown significant improvement in performance compared to other approaches. Another notable work is by Ciresan et al. [CMS12], who demonstrated improved records on MNIST [Den12; LeC+98], Latin letters [Gro95], Chinese characters [Liu+10], traffic signs [Sta+11], NORB [LHB04] and CIFAR10 [KH09] benchmarks using deep CNNs. A key question to our work is whether the success deep CNNs have had in other computer vision tasks also applies to the detection of lung nodules in CT scans and, in general, to the analysis of medical images.

Figure 2.4 illustrates an example of a CNN architecture which has the goal of predicting whether the input image contains a robot or not. It can be divided into two phases, namely

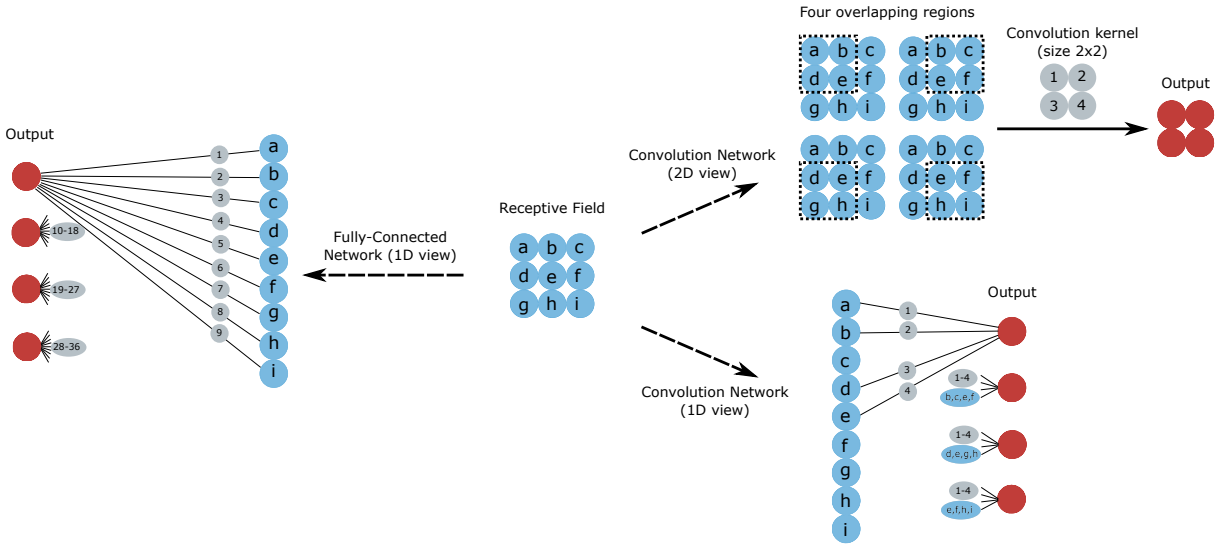

Figure 2.3: An illustration of the limited connectivity in Convolutional Neural Network and how it differs from the connectivity in fully-connected neural networks. The letters a-i represent the input pixels of the receptive field, and the numbers represent the weights (learnable parameters) of each of the networks. To the left of the receptive field, we see a fully-connected neural network which has 4 output units and a total of 36 weights (9 weights per output unit); to the right of the receptive field, we see a CNN with 4 output units and a total of 4 weights which are applied to 4 overlapping regions of the receptive field. The 2D and 1D views of the CNN are equivalent.

feature extraction and classification. The feature extraction phase is composed of two convolution layers each of which is followed by a subsampling layer, and the classification phase is composed of a flattening of the feature extraction output phase, followed by fully connected hidden and output layers. The convolution layers, as well as the fully-connected hidden and output layers, are followed by a non-linearity operation (an activation function) such as a Sigmoid, a Hyperbolic Tangent (tanh), or a Rectified Linear Unit (ReLU) function. Also, Figure 2.4 assumes a sliding kernel step size (a stride value) of 1 for the convolution layers, and 2 for the subsampling layers.

Let us examine each of these operations in more detail.

As shown in Figure 2.3, a convolution layer is composed of at least one convolution kernel which is applied on overlapping regions of the receptive field. The distance between each of the overlapping regions is called the kernel step size or stride value. This distance determines

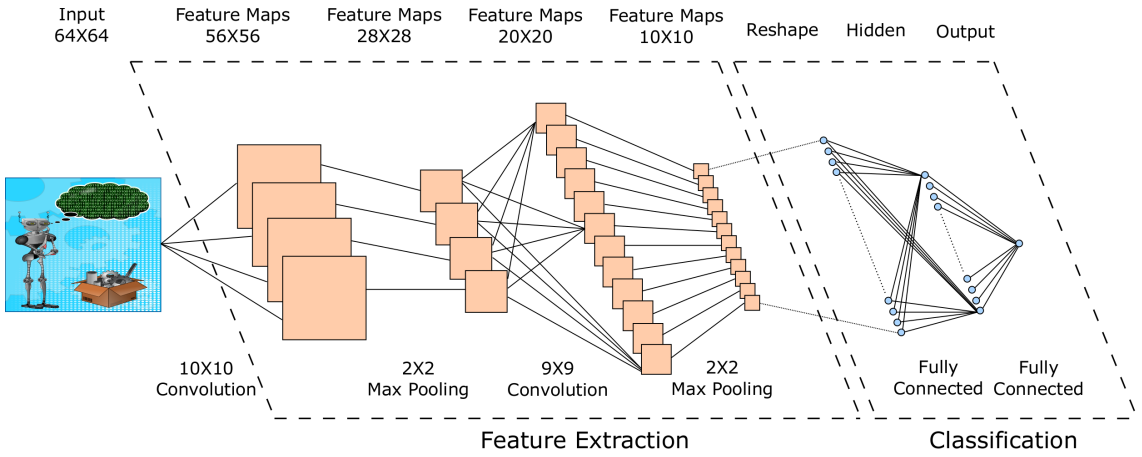

Figure 2.4: An example of a CNN architecture.

the shape of the result of the convolution operation which is called the feature map. Since convolution layers are composed of 2 dimensional (or higher dimensional) convolution kernels, the part of the network which contains these layers is considered the feature extraction phase.

More formally, performing a convolutional operation for an image that has n channels $x = \{x^c | c \in [0, n-1]\}$ each of which is of size $s \times s$, with a kernel that has n matrices $k = \{k^c | c \in [0, n-1]\}$ each of which is of size $r \times r$ $(r \leq s)$, and assuming that the stride value is 1 in both axes, results in a matrix of size $(s-r+1) \times (s-r+1)$ where each entry i, j is defined as follows:

$$(x \otimes k)_{i,j} = \sum_{c} \sum_{p,q} x^c_{i+p,j+q} k^c_{r+p,r+q} \tag{2.4}$$

where $c \in [0, n-1]$, $p \in [0, r-1]$, and $q \in [0, r-1]$.

These notations are illustrated in Figure 2.5. It shows a convolution layer with a single convolution kernel of size $r \times r \times n$, which is applied on overlapping regions of the receptive field using a stride value of 1. Assuming a receptive field of size $s \times s \times n$, the result of the convolution layer is a single feature map of size $s - r + 1 \times s - r + 1$. More feature maps can be obtained through the addition of additional convolution kernels. Notice that just as the convolution layer following the input layer considers all the channels of the input, so do subsequent convolution layers consider all the feature maps of their previous layer.

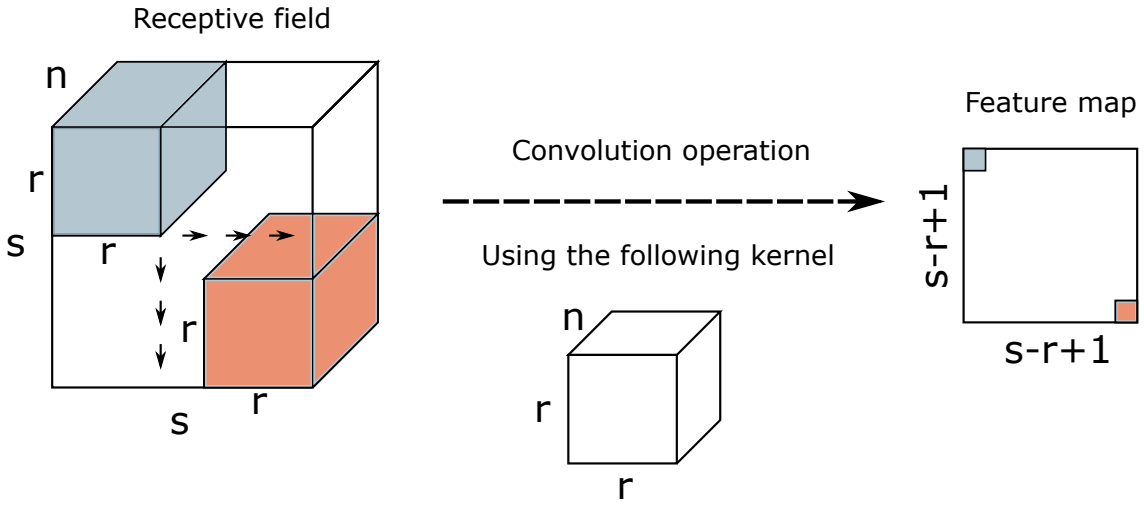

Figure 2.5: A convolution operation which converts a $s \times s \times n$ receptive field into a $s-r+1 \times s-r+1$ feature map using a single $r \times r \times n$ convolution kernel. The kernel is applied on overlapping regions of the receptive field using a stride value of 1.

A subsampling layer is usually composed of either an average pooling or a maximum pooling operation. Similar to convolution operations, pooling operations are also applied on overlapping regions of the receptive field. Consequently, pooling operations also have a stride value associated with them. The purpose of the subsampling layer is to progressively reduce the spatial size of the receptive field and therefore reduce the amount of features and the computational complexity of the network.

Performing a maximum pooling operation on a single image channel x of size $s \times s$, with a pooling kernel of size $r \times r$ ($r \leq s$), and assuming that the stride value is 1 in both axes, results in a matrix P of size $(s-r+1) \times (s-r+1)$ where each entry i, j is defined as follows:

$$P_{i,j} = max_{p,q}\{x_{i+p,j+q}\} \tag{2.5}$$

where $p \in [0, r-1]$ and $q \in [0, r-1]$

Finally, an activation layer is composed of an activation function which is applied on each of the values in the previous layer independently. In other words, the activation function is an element-wise operation. Here, we define the three activation functions we examined in this thesis, namely the logistic sigmoid, rectified linear, and hyperbolic tangent functions. These

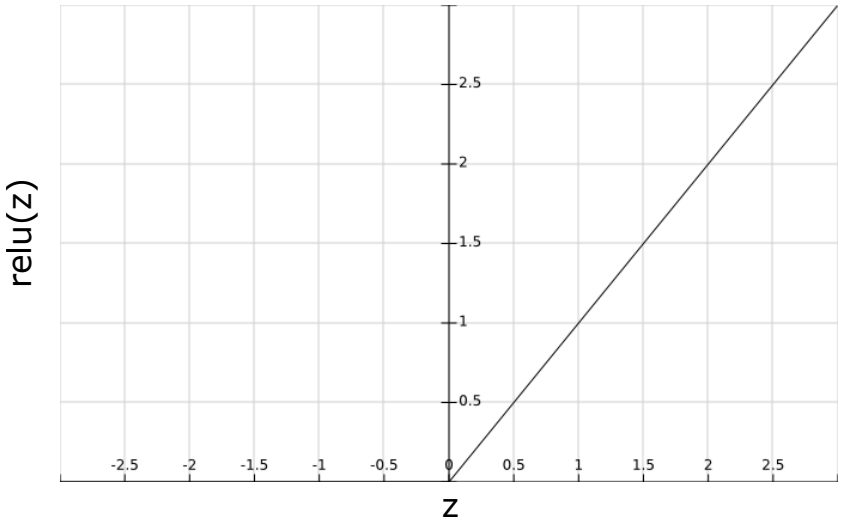

Figure 2.6: The rectified linear unit function $ReLU(z) = max(0, z)$

non-linearity operations are commonly performed after the convolution and fully-connected layers of CNNs. Figure 2.1, Figure 2.6, and Figure 2.7 illustrate the logistic sigmoid, rectified linear, and hyperbolic tangent functions, respectively.

$$sig(z) = \frac{1}{1 + e^{-z}} \tag{2.6}$$

$$ReLU(z) = max(0, z) \tag{2.7}$$

$$tanh(z) = \frac{1 - e^{-2z}}{1 + e^{-2z}} \tag{2.8}$$

To see how these three operations can be applied sequentially, Figure 2.8 shows a sequence of a convolution, ReLU, and max pooling operations. It assumes a stride value of 1 for both the convolution and max pooling operations and for both axes. Notice that the convolution operation is essentially an element wise multiplication (a discrete correlation) of the convolution kernel with the corresponding image region.

Looking at Figure 2.4 once again, we now have a better understanding of its connectivity and components, and we understand how the dimensions of the different components are

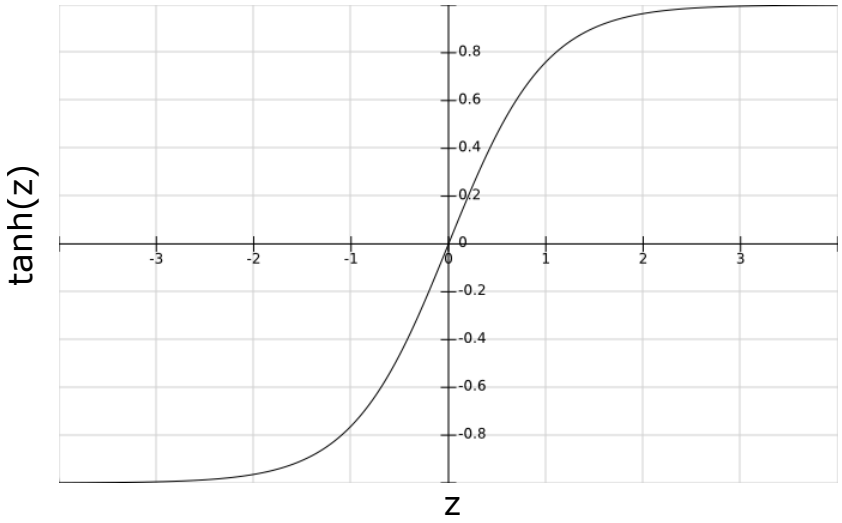

Figure 2.7: The hyperbolic tangent function $tanh(z) = \frac{1-e^{-2z}}{1+e^{-2z}}$

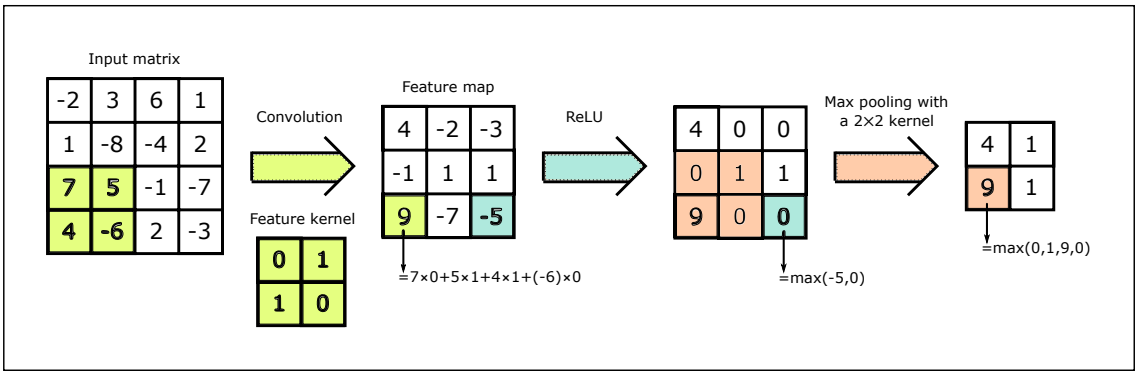

Figure 2.8: A sequence of 2D convolution, ReLU, and max pooling operations. The stride value of both the convolution and max pooling operations, along the two axes, is 1.

determined. Also, as shown in Figure 2.3, we understand how the structure of a convolutional neural network can be viewed as a neural network with limited connectivity. Once the structure of the network is determined, training can begin. In the next section, we examine the Backpropagation algorithm for learning the weights (learnable parameters) of a neural network. Once that is done, the network can be applied to perform the classification task it has been set up to do.

2.4 Learning Neural Networks with Backpropagation

Neural networks are commonly used as supervised learning algorithms for performing classification. The backpropagation algorithm [RHW88] is one of the most common learning algorithms for training neural networks. It has been suggested that using backpropagation with a relatively small batch size is preferable when it comes to training deep neural networks since it has been shown that many second order methods are impractical for large neural networks and that stochastic learning is usually much faster and often results in better solutions than batch learning [LeC+12].

To describe the backpropagation algorithm, let us first consider the following notation. As before, we assume that we have m training examples and their correct output, denoted as $\{(x^1, c^1), (x^2, c^2), ..., (x^m, c^m)\}$. Unlike before, there are K classification labels, so while x^i is still an $n+1$ dimensional vector, c^i is a K dimensional vector instead of a scalar, which has the value 1 at the index of the correct label and 0s at all other indices. We also denote L as the number of layers in the feed-forward network, and s_l as the number of units (excluding the bias unit) in layer l. θ^l denotes the matrix of all weights between layer l and layer $l+1$, and a^l denotes the activation of all the units in layer l. Finally, the symbol $*$ denotes the element wise multiplication operator, and λ is again the regularization parameter which determines how much the network is penalized for having high value weights.

Figure 2.9 shows a diagram of a feed-forward network which is using these notations. The nodes in the diagram represent the units of the network. The nodes in the first layer, denoted as a^1, represent the input of the network, and the nodes in the last layer, denoted as a^5, represent the output layer for a two class classification problem. All other layers represent the intermediate activations of the network. More specifically, node a_i^l which has incoming edges from nodes $\left\{a_j^{l-1} | 0 \leq j \leq s_{l-1}\right\}$ represents the result of the logistic function after being applied on the linear combination of all incoming nodes and their respective edges (representing the weights of the network).

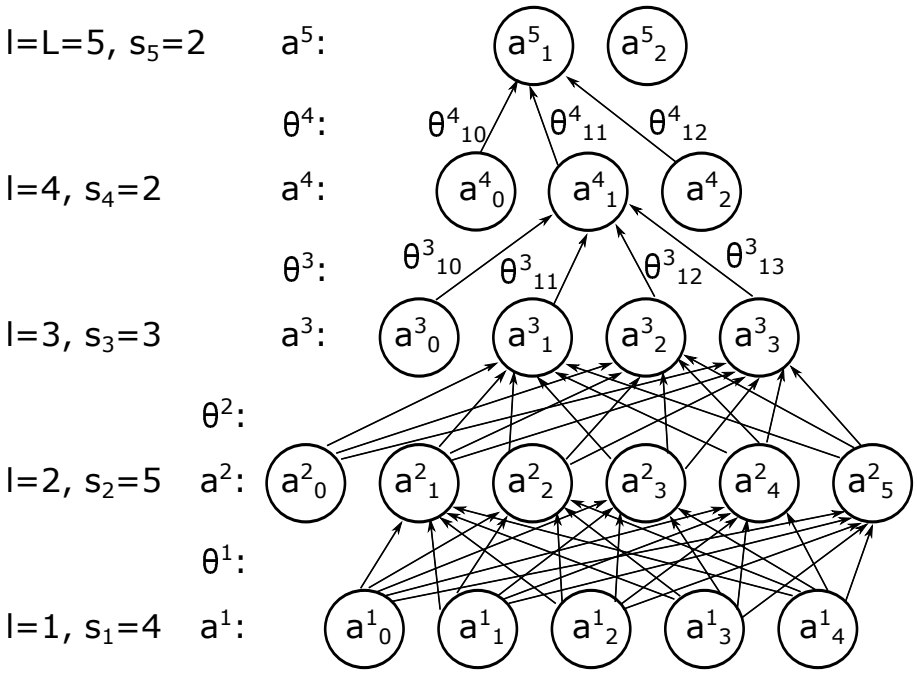

Figure 2.9: A diagram of a feed-forward network, which includes the notations that we use to describe the backpropagation algorithm. Some of the edges between layers 3 and 4, and between layers 4 and 5, have been omitted to make the diagram clearer.

More formally,

$$a_i^l = \frac{1}{1 + e^{-z}} \tag{2.9}$$

where

$$z = \sum_{j=0}^{s_{l-1}} a_j^{l-1} \theta_{ij}^{l-1} \tag{2.10}$$

Using these notations, we can now describe the loss function that the backpropagation algorithm, described in Algorithm 2, tries to minimize. The loss function is a summation of two components: a sigmoid cross entropy component and a regularization component.

$$loss(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} c_k^i \log(h_\theta(x^i)_k) + (1 - c_k^i)\log(1 - h_\theta(x^i)_k)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L+1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\theta_{ji}^l)^2 \tag{2.11}$$

where θ_{ji}^l is the weight between the i'th unit in layer l and the j'th unit in layer $l+1$ (Figure 2.9), and $h_\theta(x^i)$ is equal to the activation of the last layer of the network (a^L) after

Algorithm 2 Backpropagation

Randomly initialize θ_{ij}^l for every l, i, j
while not reaching some stopping criterion **do**
  Set $\Delta_{ij}^l = 0$ for every l, i, j
  for $i = 1$ to m **do**
    Set $a^1 = x^i$
    Compute $a^2, a^3, ..., a^L$ by applying the forward propagation rule:
    $a^l = sig(\theta^{l-1} a^{l-1})$ where $sig(z) = \frac{1}{1+e^{-z}}$
    Compute $\delta^L = a^L - y^i$
    Compute $\delta^{L-1}, \delta^{L-2}, ..., \delta^2$ by applying the backpropagation rule:
    $\delta^l = (\theta^l)^T \delta^{l+1} * a^l * (1 - a^l)$
    Update $\Delta_{ij}^l = \Delta_{ij}^l + a_j^l \delta_i^{l+1}$ for every l, i, j
  end for
  Compute $D_{ij}^l = \frac{1}{m} \Delta_{ij}^l$ for every l, i, j and add $\lambda \theta_{ij}^l$ if $j \neq 0$
  Update $\theta_{ij}^l = \theta_{ij}^l - \alpha D_{ij}^l$ for every l, i, j (gradient descent step)
end while

performing the forward propagation of x^i. This means that $h_\theta(x^i)$ is a K dimensional vector, and we denote its k'th value as $h_\theta(x^i)_k$.

Note that in computing $h_\theta(x^i)$, we assume that all the nodes in the network have the logistic function as their activation function. The logistic function is illustrated in Figure 2.1. When this assumption is made, the neural network can be regarded as an extension to the logistic regression algorithm we discussed in Section 2.2. There are various other activation functions that can be used in neural networks, and these functions can vary across the different nodes of the network. Here, we ignore such specialization in order to make our notation simpler.

The backpropagation algorithm is essentially an iterative algorithm for computing $\frac{\delta}{\delta \theta_{ij}^l} loss(\theta) = D_{ij}^l$ for every i, j, l, using these derivatives to update the weights of the network, and repeating this process until some stopping criterion is reached, such as correctly classifying all the training examples. A new data example can then be classified by performing forward propagation and selecting the label for which its vector is the closest to the activation of the last layer of the network.

When backpropagation was first applied to neural networks by Rumelhart et al. in 1988 [RHW88], it was expected that it will enable the training of networks with a large number of hidden layers, and that the units in these hidden layers will model high-level features of the raw data. It turned out not to be the case, and most of the neural networks that were trained ever since were fairly limited in the number of hidden layers they had. This trend continued to the 1990's, when neural networks were rarely used, and other learning algorithms, such as support vector machines, were favored by the scientific community. This had happened for two main reasons: (1) the computational power needed to perform the training of deep neural networks was overwhelming to the computing resources that were available at that time, and (2) the size of available datasets at that time was too small.

Nowadays, advancements in GPU technology allows artificial intelligence researchers to train deep architectures significantly faster compared to using CPU alone. The extent by which speed up is achieved depends on the hardware that is used and the type of experiment being examined. In general, utilizing the GPU is more beneficial when applied on larger network and larger batch sizes. This is done by the utilization of massively parallel graphics processors which accelerate operations that are also parallel in nature. One such operation is matrix multiplication. Computing a single layer of a neural network from its previous one in the case of forward propagation, and from its subsequent one in the case of backward propagation, can be formulated as a matrix multiplication operation. Consequently, speeding up this operation which is fundamental to the training and execution of neural networks has a significant impact on the overall computation time of the entire process. As a result, much of the state-of-the-art research that is being done in the field of deep learning is executed over GPUs.

DNNs tend to overfit small datasets due to their strong representational power. As a result, large-scale datasets with labeled data are essential for the successful training of such deep architectures in a supervised manner. Fortunately, large-scale datasets from various

domains have become increasingly available in recent years. The clearest example we see comes from the adoption of the internet by the masses and the easy access people have to digital cameras, which led to huge amounts of video and image data being shared every day by millions of people through platforms such as YouTube [You] and Facebook [Fac].

These advancements in technology, which led to an increase in computational power and allowed the generation of large-scale datasets, have led to the outstanding success that deep neural networks have shown in recent years across numerous application domains. These include computer vision [KSH12; Sze+14; CMS12; NHH15; Che+14a; LSD15; Le13], speech recognition [Lee+09; Hin+12a], natural language processing [CW08], recommender systems [ODS13], and dimensionality reduction [HS06].

2.5 Combining Multiple Classifiers

It is sometimes useful to combine multiple classifiers instead of using just one. But this raises an interesting question: how can we combine different classifiers to perform a single prediction at test time. Probably the most intuitive and simple approach is to perform the classification according to the majority vote of the different classifiers, which is known as the bootstrap aggregating (bagging) algorithm [Bre96]. The CADe system proposed in this thesis can easily be extended to a kind of bagging algorithm in which a number of DCNNs are used to compute the voting grid with which the location and boundary of nodule predictions are predicted.

Another approach for combining multiple classifiers is using a boosting algorithm [FSA99]. A boosting algorithm generates a linear combination of classifiers, which is then used to perform the classification of new examples at test time. They do so by assigning weights to the examples in the training set, and adjusting them in an iterative manner according to how well they were classified by the various classifiers.

At the beginning of the algorithm, the first classifier is trained with all the weights having the same value. Then, the coefficient of this classifier is determined, and the weights are modified according to how well the classifier performed on each of the examples. More specifically, if an example is correctly classified, its weight decreases, and if it is misclassified, its weight increases. This process is repeated for every additional classifier that is added to the system until some stopping criterion is reached. The linear combination of the various classifiers, defined by their coefficients, can then be used to perform classification on new examples for which their labels are unknown.

Applying a boosting algorithm to our proposed CADe system is more challenging since the input for the CNN (its receptive field) is a sub-volume of a CT scan and not the entire scan. The number of sub-volumes a scan contains depends on the size of the scan and the sub-volume dimensions, but is generally very large. In addition, one might want to build an ensemble of classifiers which have varying receptive field sizes. This would make the definition of an example, in the context of the boosting algorithm, more blurry. To solve this issue, an example, in the context of the boosting algorithm, can be defined as a single CT scan. Consequently, the linear combination of the various classifiers can be determined based on their performance for each of the CT scans.

In the context of neural networks, it appears to be hard to combine multiple large neural networks because it takes a long time to train each network, and because it is inefficient to run a lot of large neural networks at test time. However, there is an effective technique called Dropout [Hin+12b; Sri+14], which is used to both combine multiple neural networks, and perform regularization. The DCNN at the heart of our proposed CADe system can easily be trained and deployed with this technique.

The dropout algorithm can be summarized as follows. Let us assume that we have a neural network with H hidden units, spanned across multiple layers. Each time we present the network with a training example, we randomly omit each hidden unit with a probability

of 0.5. As a result, we are essentially sampling from 2^H different model architectures, which all share the same weights. This means that only a few of the 2^H architectures get trained, and that when they do, it happens with a small number of training examples, depending on the size of H. Also, since they share the same weights, each model is strongly regularized towards what other models want. In other words, instead of pulling the weights (parameters) of the network towards zero, which happens in the other form of regularization we mentioned above, the weights are pulled to what other models want.

Finally, we can perform forward propagation on the resulting network by halving the outgoing weights of all the hidden units, which estimates the geometric mean of the predictions of all 2^H models. It has been shown [Hin+12b; Sri+14] that if a deep neural network is strongly overfitting, Dropout has the capacity to significantly reduce its error rate.

Chapter 3

Related Work

Convolutional Neural Networks (CNNs) have shown to produce some promising results in many application domains such as speech recognition [Lee+09; Hin+12a], recommender systems [ODS13], and natural language processing [CW08], but they are most notable for their state-of-the-art results in computer vision tasks [KSH12; Sze+14; CMS12; NHH15; Che+14a; LSD15].

In this chapter we first examine how deep CNNs have been successfully applied to two well-known computer vision tasks, namely the recognition of hand-written digits from the MNIST dataset [Den12; LeC+98], and the recognition of various objects in natural images as part of the ImageNet challenge [Den+09; Rus+14]. Then, we examine a third vision task, namely the detection of lung nodules in thoracic CT scans, and explore some of the work that has been done to tackle it. While doing so, we examine the various differences and similarities between our work and the work of others, and give the reader some context for the following chapters where we dive deeper into the specifics of our work.

3.1 Natural Image Classification

The main goal of natural image classification algorithms is to allow computers to determine the content of images that humans are exposed to in their daily lives. They do so by a form of signal processing, i.e. applying mathematical operations on the pixel values of the input image. While natural image classification is somewhat different from the analysis of CT scans, we still explore it here because of the tremendous results that Deep Convolutional Neural Networks have demonstrated in this area in recent years.

3.1.1 MNIST

The MNIST (Mixed National Institute of Standards and Technology) database is a large database of handwritten digits, which consists of 60,000 training images and 10,000 test images [Den12; LeC+98]. Ciresan et al. [CMS12] were able to produce the state-of-the-art results for this task using a committee of 35 deep CNNs, where each network is trained on inputs that are preprocessed in different ways, and their outputs are averaged. During training, the input images of each CNN are continually translated, scaled, elastically distorted, and rotated, whereas only the original images are used for validation. Using a single deep CNN by itself already yields better results than the state-of-the-art for this task, but using a committee of CNNs further decreases the error rate by 30-40%.

One interesting property of this CNN is that its convolution kernels have shown to be most beneficial when they are small and even minimal. This means that even though the feature detectors (convolution kernels) throughout the entire network are small, together they still have the capacity to capture global features very well. One explanation for this is as follows. As the computation of the network moves from the input layer to the output, downsampling operations such as pooling layers decrease the size of the receptive field and therefore allow for convolution kernels that are further away in the computation to capture more global features of the input.

Another interesting observation is that the network architecture applied by Ciresan et al. for the MNIST task has also demonstrated improved records on a number of other computer vision benchmarks such as the recognition of Latin letters [Gro95], Chinese characters [Liu+10], traffic signs [Sta+11], NORB [LHB04], and CIFAR10 [KH09]. This is particularly interesting since there were no carefully pre-wired units (artificial neurons) in those CNNs, which means that no expert knowledge was incorporated in any of these different application domains. It is true that all these tasks are from the computer vision domain, and therefore are assumed to be of similar nature, but each also has its own unique properties. Indeed,

it is exciting that a single network architecture is able to show high performance on various computer vision tasks. However, one needs to be careful when concluding that this will also be true across more distant application domains.

The successful use of continuous distortions of the original images during training is essentially a way to increase the size and variability of the training set. This has shown to be important for the training of deep CNNs because of their large number of parameters, which result in their tendency to overfit small training sets that have little variability. Of course, such generation of input examples is not necessarily possible in other application domains, but at least in the case of MNIST, we can see that deep CNNs become advantageous with increased size and variability of their training set. Such work is also suggestive that unsupervised initialization/pretraining is not necessary when the training set is of sufficient size and variability.

Another technique for overcoming overfitting, in addition to increasing the size and variability of the dataset, is using Dropout [Hin+12b; Sri+14]. The key idea of Dropout is to randomly drop units (along with their connections) from the neural network during training, which essentially means that the training is performed on an exponential number of "thinned" neural networks. During testing, the effect of all these networks is estimated by using all neural units and halving their weights. This prevents units from co-adapting too much, and therefore, reduces overfitting and improves upon other regularization methods. We have discussed this topic more thoroughly in Chapter 2.5.

3.1.2 ImageNet

ImageNet [Den+09; Rus+14] is a publicly available database and an annual online contest (started in 2009), where one of its associated challenges is to classify more than 1.4 million high-resolution images into 1000 classes (representing the various objects contained in the images). Similarly to MNIST, large-scale CNNs have shown to produce the state-of-the-art results for this task [KSH12; Sze+14].

The experiments in [KSH12] suggest that better results can be obtained by simply utilizing faster GPUs and larger datasets. Alternatively, Szegedy et al. [Sze+14] have shown that significantly more accurate results can be achieved by modifying the architecture of the network even if it uses 12 times fewer parameters than the network proposed in [KSH12]. Having a smaller network improves the efficiency of the algorithm in terms of power and memory usage, and allows it to be deployed in more devices such as mobile and embedded computers.

In contrast to MNIST, NORB, and CIFAR10, which contain an order of tens of thousands of labeled images, ImageNet has millions of labeled images. Pinto et al. [DPC08] have recognized the shortcomings of small image datasets. Their work suggests that the performance of any classifier significantly depends on the real-world image variation in the datasets they are trained on. In other words, one has to be careful when evaluating the generality of his proposed model, especially if the dataset used is small and has little variability. This is because objects in realistic settings usually exhibit considerable variability.

Seeing the success that CNNs have shown in the above computer vision tasks, it is interesting to examine whether CNNs can also be successfully applied to the detection of lung nodules in thoracic CT scans. This is a key questions in this thesis, and in the next sections we will explore some of the work in this area of research.

3.2 The detection of lung nodules in thoracic CT scans

In a comprehensive survey conducted by Suzuki et al. [Suz13], three classes of classification techniques, into which the various CADe systems can be categorized, are identified. These are (1) Feature-based Machine Learning (FML), (2) Pixel/voxel-based Machine Learning (PML), and (3) Non-ML-based methods. While some non-ML methods are briefly discussed in this chapter, we focus on the first and second classes of classification techniques since these are more related to our work.

3.2.1 Feature-based Machine Learning

The work described below [Ric+11; Gol+09; MHR10; Tan+11] belongs to the FML class of classification techniques since it utilizes features that are hand-engineered by the designers of the system. These CADe systems have used the LIDC-IDRI dataset for performance evaluation, and therefore we can compare our results to theirs. Another commonality among these approaches is that they are comprised of three steps: (1) lung segmentation, (2) preliminary detection of candidate lung nodules, and (3) a false positive reduction step.

In the following paragraphs, we focus on the false positive reduction step since it is most related to our work. There is a range of techniques which have been used to perform the first and second steps [TM+08; AB94; LZ03]. These include various filtering techniques, image transforms, and region growing techniques, but in general, they usually rely on some geometric assumption about the objects they aim at detecting and segmenting, whether it is a lung, a blood vessel, or a lung nodule. For example, in order to detect a nodule candidate, there is usually the assumption that the nodule has a radial symmetry, and therefore techniques such as 3D fast radial filtering are performed [LZ03]. Alternatively, in order to segment the lung's boundary, it is assumed that these boundaries are well defined in CT scans, and therefore, a combination of histogram thresholding, seeded region growing, and mathematical morphology are applied [TM+08; AB94].

3.2.1.1 Computer-aided detection of lung nodules via 3D fast radial transform, scale space representation, and Zernike MIP classification [Ric+11]

After a set of CT regions which contain candidate lung nodules is obtained, and a simple heuristic false positive reduction step is performed, which discards CT regions that are not in the appropriate diameter range of 3mm to 30mm, Riccardi et al. [Ric+11] use a supervised false positive reduction step based on Maximum Intensity Projection (MIP) filters [Gru+02] and Zernike moments [KH88] to return the final nodule predictions. This step can be summarized in four processing steps: (1) volume of interest (VOI) cropping and resizing,

(2) MIP processing in three directions, (3) feature extraction, and (4) SVM classification.

In step 1, the CT regions are resized to a cube by means of linear interpolation. In step 2, three MIP filtered images are obtained for each of the CT regions by ray projection techniques, where the value assigned for each ray is the maximum value encountered by the ray along its path. The logic behind using MIP filtering is that blood vessels and nodules have very similar 2D cross sections in thin slice CT scans, but can still be distinguished since nodules remain circular in shape, while vessels are seen as elongated strips when examining the three MIP images. This property raises a concern about the applicability of this filtering technique when facing elongated lung nodules or when a low-dose CT scan is being analyzed where the CT slice thickness is large. For example, analyzing CT scans with a high slice thickness might lead to a decrease in the true positive rate, since nodules extracted from such images might appear as elongated stripes and therefore be wrongly predicted as vessels.

Final features are obtained by computing the rotation invariant Zernike moments [KH88] from the MIP images (step 3). While these features are also insensitive to slight deviation in the nodule structure, their main use is due to their rotation invariant properties. Riccardi et al. argue that these sets of features are particularly appropriate to lung nodules since these usually appear as "perturbed disks" in CT scans. However, this assumption might not always hold true since lung nodules can have other complex structures.

Finally, these features are used to train a support vector machine (SVM) classifier (step 4), which predicts whether each CT region contains a lung nodule. A SVM classifier can be viewed as a shallow fully-connected neural network, where the weights between the input layer and the hidden layer represent the kernel function. It would be interesting to examine whether a "deeper" classifier, i.e. a classifier with multiple non-linear transformations such as a NN with multiple hidden layers, can improve the performance of such a system.

3.2.1.2 A novel multithreshold method for nodule detection in lung CT [Gol+09]

After segmenting the lungs and selecting ROIs based on a multi-threshold surface-triangulation approach, Golosio et al. [Gol+09] extract features from these ROIs which are used as input to a NN classifier. The features include measures of volume, roundness, maximum density, mass, and principal moments of inertia, which are computed based on the triangulated model of the ROI. Many of the ROIs are generated due to noise in the image, so these features are expected to enable the discernment between nodules, noise, and other objects such as blood vessels.

The logic behind using these features is that, again, nodules are considered to be spherical by nature, so their roundness measurement is expected to be close to 1 and their three principal moments of inertia are expected to be similar. More specifically, the volume and density of ROIs that originate from noise is much lower than that of nodules. In regards to blood vessels, their volume can be very large, but their roundness is generally lower than that of nodules, and the principal moment of inertia corresponding to the axis parallel to the vessel is generally much smaller than the other two.

Golosio et al. have defined these features as a function of the threshold used to obtain the triangulated model of the ROIs, so it is argued that, for example, the behavior of the volume and roundness functions is different for isolated nodules compared to nodules that are connected to blood vessels. More specifically, while the roundness value is high in all thresholds for isolated nodules, it shows a peculiar behavior for nodules that are connected to blood vessels. For low thresholds, the roundness value is quite low, presumably since the nodule is connected to the vessels and therefore is part of a big ROI which include the long vessels; but as the threshold increases, the nodule separates from the vessels and the ROI volume rapidly decreases and at the same time the roundness will rapidly increase.

Once the lung segmentation, ROI selection, and feature extraction steps are done, the features are used as input to a relatively small neural network with a single hidden layer,

where the input layer size is 43 (seven features evaluated for six threshold values, plus the maximum density inside the ROI), the hidden layer size is 11, and the output layer size is 2, corresponding to a positive and negative response.

It is worth mentioning that the multi-threshold surface-triangulation approach used by Golosio et al. was not very selective, meaning that the number of ROIs obtained is very large, producing a much greater number of negative examples than positive ones. Consequently, they used a Self Organizing Map (SOM) to downsample the negative examples with the hope of not altering their distribution in the feature space. More specifically, they trained a SOM with an output layer size of 10x10 using an unsupervised algorithm, used it to associate each of the ROIs with one of the output nodes, and then downsampled the negative examples by taking about 2% of the cases from each output node of the SOM. Downsampling can also be performed by other unsupervised learning algorithms such as the k-means clustering algorithm or the Persistent Contrastive Divergence (PCD) algorithm [Tie08] for learning a stack of restricted Boltzmann machines.

3.2.1.3 A new computationally efficient CAD system for pulmonary nodule detection in CT imagery [MHR10]

The CADe system of Messay et al. [MHR10] is also an interesting one. They first preprocessed the data, which included orienting and down-sampling the data in order to generate CT slices with a comparable orientation and slice spacing, and performed local contrast enhancement in order to improve the details and local context of lung nodule candidates. Then, they performed lung segmentation followed by 3D nodule candidate detection and segmentation based on multiple gray level thresholding and a simple size and compactness based expert filter.

Once this is done, a set of 245 2D and 3D features is extracted from each segmented nodule candidate. This set includes geometric, intensity and gradient features. An extensive list of all 245 features can be found in [MHR10]. The 2D geometric features represent the shape

and position of the candidate nodules in relation to the center of the lung, and are evaluated based on the largest area slice of the nodule candidate segmentation. These include size, circularity, and distance to the center of the segmented nodule candidate. The 3D geometric features include measurements such as elongation, cube compactness, and fraction touching the lung. Furthermore, 2D and 3D intensity features include minimum and maximum values inside and outside of the segmented nodule candidate, standard deviation inside and outside of the segmented nodule candidate, and contrast. Finally, 2D and 3D gradient features include radial-deviation and radial-gradient statistics inside and outside of the segmented nodule candidate. 3D features also include features above and under the segmented nodule candidates in order to represent the tissue that surrounds the candidate nodule.

A feature selection algorithm is then applied to determine two subsets of the 245 candidate features to be used for two distinct classifiers with the objective of maximizing the area under the FROC curve. Messay et al. used two simple classifiers, namely a Fisher Linear Discriminant classifier [Fis36] and a quadratic classifier. The quadratic classifier, being a generalization of the linear one, is expected to be able to represent more complex separating surfaces between positive and negative candidate nodules, but has shown to be inferior to the linear classifier. This can be due to an overfitting effect, caused by the relatively small number of training samples (90 thoracic CT scans), and of true positive samples in particular.

3.2.1.4 A novel computer-aided lung nodule detection system for CT images [Tan+11]

Finally, Tan et al. [Tan+11] have demonstrated a CADe system with somewhat superior results to the above systems. It can be described as follows. First, re-sampling of the DICOM images to a fixed slice thickness using Tri-linear interpolation is performed, followed by a lung segmentation procedure. Then, the system applies a nodule candidate detection algorithm which considers three different kinds of nodules, namely isolated, juxtavascular (or vessel-connected), and juxtapleural (or pleura-connected) nodules, and is based on a set of selective nodule and vessel enhancement filters. The main problem of using these filters is the high

amount of FP detections they produce, especially in the locations of vessel branches and junctions. In response, Tan et al. employ a nodule center estimation procedure, a grey-level thresholding procedure, and a region growing algorithm to reduce the number of FPs. Then, they cluster overlapping nodule segmentations, which further reduces the number of FPs.

Once a set of segmented candidate nodules is obtained, feature extraction is performed. Tan et al. uses two kinds of invariant features, namely features that are based on the isophotes of candidate nodules, and other classical shape and grey-value descriptors. Isophotes are lines drawn through areas of constant brightness, and are similar to contour maps which show lines through areas of constant elevation. Isophotes are invariant to the orthogonal group of spatial transformations and the group of general intensity transformations, which makes them appropriate to the analysis of CT scans.

Consequently, Tan et al. used isophote-based features such as a ridge detector, isophote curvature, a measure for isophote density, a measure of deviation from flatness, a checker-board detector, and a Y-junction detector. The logic behind using such features is to discriminate between the presence of spherical lung nodules and the presence of tubular structures like blood vessels. In addition, the set of classical geometric and grey-value descriptors Tan et al. have used includes features such as volume, compactness, elongation factor, and distance of nodule candidate centroid to lung wall. A complete list of these features can be found in [Tan+11].

Once feature extraction is done, Tan et al. employ three different classifiers and compare their results. These include (1) a feature-selective classifier based on NNs and genetic algorithms (FD-NEAT) [Tan+09], (2) SVMs, and (3) fixed-topology NNs. FD-NEAT is a machine learning algorithm to automatically discover the topology and weights of neural networks by means of a genetic algorithm. More specifically, evolution starts with a minimal network topology, where all the input units are connected to the output units, and structure is added incrementally through the mutation operators. FD-NEAT is very similar to

the original NEAT system [SM02], except that it has some extra mutation operators that allows for the pruning of input features, thus making it a feature selection algorithm as well. The performance of the three classifiers varies depending on their configuration at different agreement levels, but in general the overall performance of the NN-based classifiers slightly exceeds that of SVM.

3.2.2 Pixel/voxel-based Machine Learning

Similarly to our CADe system, the works mentioned below [Ani+16; Shi+16; Bus05; GG16] belong to the PML class of classification techniques. This means that the spacial/volumetric feature detectors they contain are trained in a supervised manner from the input data, i.e. from the pixel/voxel values of chest radiographs/CT images. We are one of the first research groups to utilize Deep Convolutional Neural Networks (DCNNs) for the detection of lung nodules in CT scans [GJD16]. When we first approached this task, which was at the beginning of the year 2015, we were not able to find any publication which applied DCNNs for this task. However, throughout the year 2016, there have been a number of publications [Ani+16; Shi+16; Bus05; GG16] which applied DCNNs to tackle other similar tasks; and in 2017, a number of survey papers [Lit+17; SWS17] have reviewed the application of deep learning in medical image analysis.

3.2.2.1 Lung nodule detection using 3D convolutional neural networks trained on weakly labeled data [Ani+16]

Anirudh et al. [Ani+16] have used a DCNN which has a very similar architecture to the one we used in [GJD16]. Instead of using a fully annotated dataset such as the LIDC-IDRI dataset, which includes a complete segmentation of all the nodules it contains, they used a single point and a largest expected size to represent each nodule. They implemented an unsupervised segmentation algorithm to grow out each 3D region, which is used during both the training and testing of their CNN. While this approach indeed makes the annotation

process easier, it still requires a radiologist to annotate the data. Having intuitive software tools for radiologists, which allow them to easily segment nodules on a PC or tablet, would still achieve the same goal while not compromising the quality of the annotations. Lung nodules do not always appear to be continuous objects in CT scans, and as a result, an algorithm that grows out a 3D region on such nodules can fail to capture the entire nodule.

In order to assess the performance of this unsupervised grow out algorithm, one should only use it to segment the nodules in the training set while using fully annotated nodules in the test set. Anirudh et al. have not done so, and instead, they used the unsupervised algorithm to segment lung nodules in both the training and test sets. Consequently, we consider their results to be somewhat misleading and biased. Also, they only used 67 CT scans from the SPIE-AAPM-LUNGx dataset in their work, which we consider to be insufficient for producing results without any model validation technique such as k-fold cross-validation, even though they used a training set of size 20 and a test set of size 47.

Finally, similarly to our approach, Anirudh et al. have not used any lung segmentation tool, but they do acknowledge that it is indeed not a trivial thing to do and that doing so properly should help in reducing the false positives of any model. In our work, we tried to avoid using a lung segmentation procedure for a similar reason, namely the challenge of segmenting the lungs and the effect a wrong segmentation might have on the detection of lung nodules. Anirudh et al. [Ani+16] did not use the LIDC-IDRI dataset to evaluate the performance of their models, and therefore we cannot compare our results to theirs.

3.2.2.2 Other related work [Shi+16; GG16; Bus05]

Other work worth mentioning, where DCNNs have been used but on slightly different tasks, include [Shi+16; GG16; Bus05]. Such work can not be compared to our work since it does not utilize the LIDC-IDRI dataset or it does not describe a CADe system for the detection of lung nodules in complete CT scans.

Shin et al. [Shi+16] studied two specific CADe problems, namely thoraco-abdominal lymph node detection and interstitial lung disease classification. While they did not look at the detection of lung nodules, detecting other abnormal structures in the lung is very much related, and is supportive to the idea of using DCNNs to detect lung nodules in thoracic CT scans.

Gruetzemacher et al. [GG16] used DCNNs for the binary classification of lung nodules rather than the detection of lung nodules in complete CT scans. More specifically, they do not present a CADe system which has the capacity to detect lung nodules in complete CT scans. Instead, their system only predicts whether a given volume contains a lung nodule or not. To do so, they rely on sets of candidate nodules generated using existing CADe systems, which can lead to evaluation problems since errors from these existing CADe systems will propagate to the evaluation of the binary classification algorithm.

Finally, Bush [Bus05] has used DCNNs to detect malignant lung nodules in chest radiographs, which is interesting since chest radiographs lead to much less radiation exposure for the patient, and therefore are preferable. Being able to accurately detect lung nodules based on radiographs should lead to a reduction in cancer incidences caused by medical imaging radiation exposure. However, as mentioned in 1.2, there is a statistically significant 20.3% relative reduction in lung cancer mortality when using low-dose helical computed tomography scans as a screening modality compared to using chest x-ray [Kra+11], which emphasizes the limitations of chest radiographs.

Chapter 4

DeepCADe: A Novel Computer-Aided Detection System for Lung Nodules in Thoracic CT Scans

In this chapter, we describe the overall architecture of DeepCADe, a novel CADe system for the detection of lung nodules in thoracic CT scans. We discuss the preprocessing steps that are required to construct this system, and examine the various meta-parameters that are part of its configuration. DeepCADe is based on a Deep Convolutional Neural Network (DCNN) which has been trained to detect lung nodules in sub-volumes of CT scans. The DCNN is used to compute a 3D voting grid, which has the same size as the examined CT scan, and with which the location and boundary of lung nodules are predicted. Finally, we discuss the use of a Free-Response Receiver Operating Characteristic (FROC) curve as a method for evaluating the performance of CADe systems.

Figure 1.2 illustrates a high-level diagram of DeepCADe. It shows how a single network is applied to two overlapping receptive fields (sub-volumes of a CT scan), and how the output of the network is aggregated into a single voting grid, which describes the predicted probability of a nodule in each location in the original scan.

4.1 Preprocessing

As mentioned in Section 1.3, the scans in the LIDC-IDRI dataset have been examined by four experienced thoracic radiologists. Their annotations are independent of each other so one has to implement a grouping procedure, which will determine which nodule annotations represent the same lung nodule and which are not. This is illustrated in Figure 1.4. A wrong grouping procedure will lead to an incorrect validation of any CADe system.

Here, we propose a simple and effective grouping procedure, which leads to a one-to-one correspondence with the nodule-count-by-patient file that was recently released by the LIDC-IDRI team [Tea15]. This file contains "ground-truth" information regarding the number of nodules ≥ 3 mm that are found in each of the 1018 CT images, and is based on a subsequent shared analysis of a group of radiologists. Algorithm 3 describes our grouping procedure.

Algorithm 3 Annotation grouping procedure

Let $GP[s]$ be the number of nodules in scan s as described in the nodule-count-by-patient file [Tea15]
Let $A[s]$ be the centers of all nodule annotations made by four experienced radiologists on scan s
Let $AN[s]$ be the number of nodule annotations in scan s
for $s = 1$ to $number\_of\_scans$ **do**
    while $AN[s] > GP[s]$ **do**
        Identify all nodule annotation pairs in $A[s]$ in which the annotations are from two different radiologists
        For each pair, compute the distance between the center of its constituting annotations
        Pick the pair with the smallest distance and group its constituting annotations into a single annotation
        Set the center of the new annotation to be the center of the union between the two original annotations
        Subtract $AN[s]$ by 1
    end while
end for

The authors in [Ric+11; Gol+09; MHR10; Tan+11] did not have this file available to them at the time their work was conducted, and therefore they had no ground-truth information with which they can validate the quality of their grouping procedures. Their grouping procedures are based on either the distance between centers of nodules or the overlap between nodules, but in any case, they had to incorporate a distance or overlap threshold to determine which nodule markings are grouped together and which are not. This variability in grouping procedures poses a challenge when trying to compare the performance of various CADe system. In order to mitigate this variability, the nodule-count-by-patient file [Tea15] can serve as a ground truth for the number of nodules that are present in each CT

Table 4.1: Properties of the LIDC-IDRI dataset at four agreement levels.

Agreement level	totalNod	maxNodPerScan	avgNodPerScan
1	2670	23	2.62
2	1886	13	1.85
3	1395	12	1.37
4	908	8	0.89

scan. Consequently, we use the nodule-count-by-patient file in our work to allow for a more accurate performance comparison in future research.

Once the grouping of nodule markings is complete, we can associate each nodule with one of four agreement levels. Agreement level j, where $1 \leq j \leq 4$, includes all those nodules which were marked by at least j radiologists. This is illustrated in Figure 1.4. We expect nodules at a higher agreement level to be more easily detectable by a CADe system since they were identified by more radiologists. Table 4.1 describes, for each agreement level, the total number of nodules in the dataset (totalNod), maximum number of nodules per scan (maxNodPerScan), and average number of nodules per scan (avgNodPerScan). The minimum number of nodules per scan for all agreement levels is zero.

4.2 Training a DCNN

The DCNN which is at the core of DeepCADe is trained using the backpropagation algorithm [RHW88] to detect lung nodules in CT image sub-volumes of various sizes. We chose backpropagation with a relatively small batch size, since it has been shown that many second order methods are impractical for large neural networks and that stochastic learning is usually much faster and often results in better solutions than batch learning [LeC+12].

The input of our proposed DCNN is composed of:

1. a sub-volume of a CT image which is of a pre-determined size, i.e. the receptive field of the network,

2. three values representing positional information of the sub-volume in relation to the entire CT image and for each of the three axes, and

3. four values representing information regarding the DICOM image, namely slice thickness (in mm), pixel spacing in each of the two in-plane axes (in mm), and the image orientation.

The output of the DCNN is a value in the range $[0, 1]$, representing its estimate to whether the sub-volume contains a lung nodule or not.

The DCNN is composed of two modules. The first module is designed to extract valuable volumetric features from the input data, and is composed of multiple volumetric convolution, rectified linear units (ReLU), and max pooling layers. The second module of the DCNN is the classifier. It is composed of multiple fully connected layers and non-linearity operations, and is expected to perform the high-level reasoning of the neural network. Figure 4.1 illustrates a simple CNN which has similar layers to those used in our experiments. Each color represents the use of a different 3D feature kernel, and the equivalence symbol which appears after the max pooling layer represents a flattening (reshaping) of all the values in the previous layer into a one dimensional vector. A detailed description of these operations can be found in Section 2.3, which includes an illustration of the limited connectivity in Convolutional Neural Network and how it differs from the connectivity in fully-connected neural networks (Figure 2.3).

The architecture of a DCNN can be described by listing its layers in sequence from input to output. For example, here is one DCNN architecture we explored in this thesis (denoted as $e3$): $5\times20\times20-128C3\times9\times9-MP1\times2\times2-256C2\times4\times4-MP2\times2\times2-512C1\times3\times3-512C1\times3\times3-MP1\times2\times2-2709FC-2709FC-2709FC-1FC$, where, for example, $5\times20\times20$ is the input layer (the receptive field of the DCNN), $128C3\times9\times9$ represents a volumetric convolution layer with 128 feature kernels of size $3\times9\times9$, $MP1\times2\times2$ represents a max pooling layer with a kernel of size $1\times2\times2$, and $2709FC$ represents a fully-connected

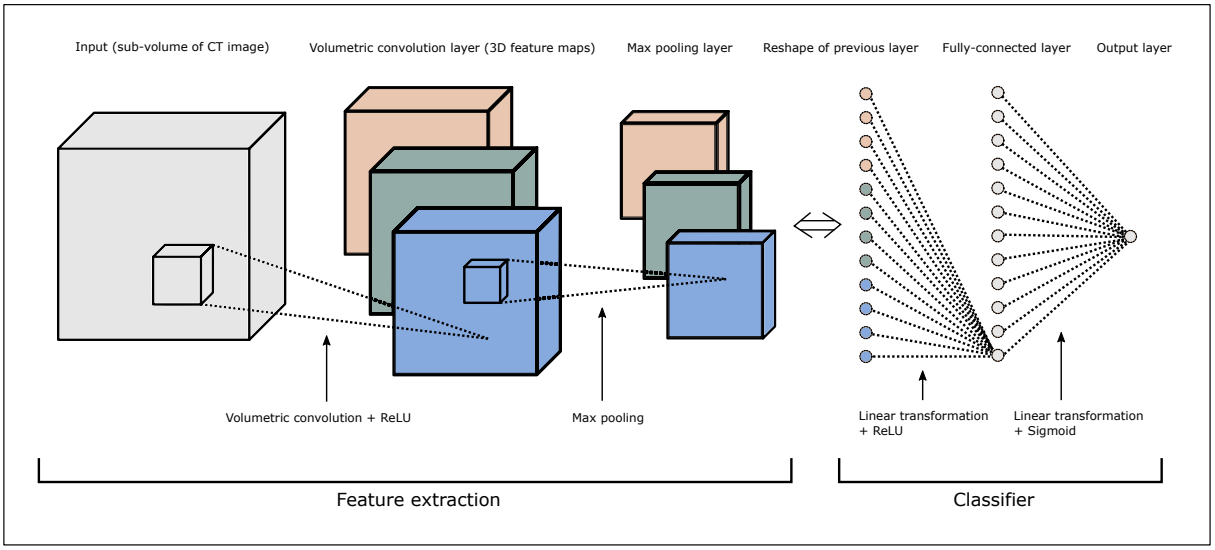

Figure 4.1: A simple CNN which has similar layers to those used in our experiments. Each color represents the use of a different 3D feature kernel, and the equivalence symbol represents a flattening of all the values in the previous layer into a one dimensional vector.

layer with 2709 units. Appendix A describes all the DCNN architectures we examined in our work using this notation.

Other meta-parameters that are not included in this description are: (1) the stride value of both the volumetric convolution and max-pooling layers, (2) the stride value used for computing the voting grid, (3) the activation functions that are applied after every convolution and fully connected layer, and (4) the zero padding size. We omitted these meta-parameters from the string representation of a DCNN architecture in order to make the representation more concise, and since these meta-parameters are constant across many of the experiments we conducted in our work. Instead, Appendix A contains a paragraph which summarizes these values for all the architectures we examined in our work, and Chapter 5 contains a complete description for each of our experiments.

In addition to the activations of its previous layer, the first fully-connected layer of the CNN receives 7 additional values. The first 3 values represent positional information of the input in relation to the entire CT image and for each of the three axes; the last 4 values represent information regarding the DICOM image, namely slice thickness (in mm), pixel

spacing in each of the two in-plane axes (in mm), and the image orientation. The motivation behind adding these values to the input of the first fully-connected layer was to provide the classifier with values which might help it achieve its classification goal and make it more robust to changes in the way CT images are acquired and to the relative location of its input.

These additional values have not shown to benefit the overall performance of our examined DCNN architectures, but we still include them in our examined architectures to demonstrate the capacity of DCNNs to process non-spatial input. This lack of benefit to performance can be due to a number of reasons. First, it is possible that the number of additional values, namely 7 additional values, is too small compared to the total number of units in the first fully-connected layer. Second, it is possible that the scale of these 7 values is significantly smaller compared to the scale of the other activation values of that layer. Third, it is possible that these sources of variability are not significant in the data we used for training and validation.

In Chapter 5, we explore how various DCNN architectures affect the performance of DeepCADe. We examine how the depth and size of the classifier module of the DCNN affect performance, and how multiple convolution layers can benefit the performance of DeepCADe. We examine three kinds of activation functions which are applied after every convolution and fully connected layer. These include the rectified linear unit (ReLU), hyperbolic tangent (tanh), and sigmoid functions, which are illustrated in Figures 2.6, 2.7, and 2.1, respectively.

Also, we experiment with varying sizes of receptive fields, i.e. the size of the CT subvolume which the DCNN receives as input. By doing so, we are essentially performing an empirical evaluation for the optimal receptive field size. Alternatively, one can determine the receptive field size manually based on the relationship between the receptive field of the CNN and the size of the various nodules in the dataset.

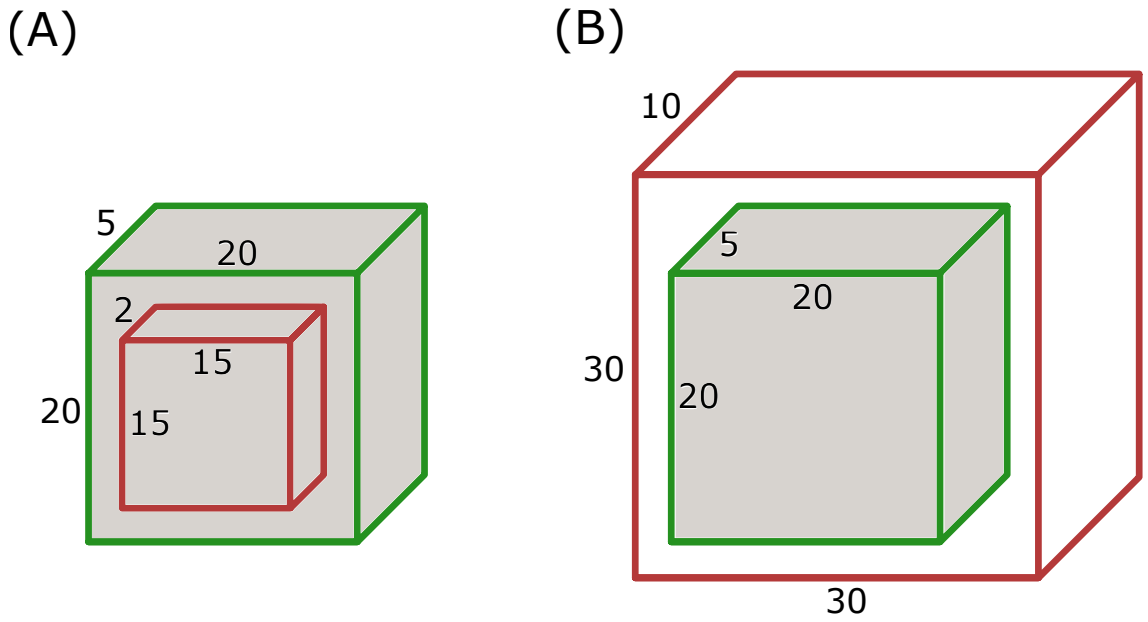

Figure 4.2: The spatial relationship between a receptive field and a nodule bounding box. The receptive field is marked by a green rectangle and the nodule bounding box is marked by a red rectangle. (A) a receptive field of size $5 \times 20 \times 20$ contains an entire nodule bounding box of size $2 \times 15 \times 15$. (B) a receptive field of size $5 \times 20 \times 20$ is contained inside a nodule bounding box of size $10 \times 30 \times 30$.

The following statistics allow the designer to perform just that. The maximum/minimum/average in-plane diameter of a nodule in the dataset is 76/1/15 voxels, respectively, and the maximum/minimum/average depth (i.e. number of slices) of a nodule in the dataset is 56/1/6 voxels, respectively. This means that a receptive field of size $5 \times 20 \times 20$, for example, can either be contained inside a nodule or it can contain an entire nodule. Figure 4.2 illustrates this spatial relationship. It shows two scenarios, where a $5 \times 20 \times 20$ receptive field is either contained inside a nodule (Figure 4.2(A)) or is containing an entire nodule (Figure 4.2(B)). The receptive field is marked by a green rectangle and the nodule bounding box is marked by a red rectangle.

Consequently, a DCNN that is based on such a receptive field of size $5 \times 20 \times 20$ is expected to learn volumetric features that are present both inside lung nodules and in their surroundings. It is important to note that in order to preserve the original values of the DICOM images as much as possible, no scaling was applied to the CT images of the dataset.

The DCNN architectures that are examined in this thesis are trained using the back-propagation algorithm, where the LIDC-IDRI dataset is used to generate the training and validation sets. We generate these sets by using a 90/10 ratio to randomly split the CT images of the LIDC-IDRI dataset. This leads to a training set of size 916, and a validation set of size 102, out of the 1018 images of the dataset. Having an independent validation set ensures that the validation results of our CADe system are good indicators for its generality and robustness. We chose a 90/10 ratio to split the data since it allowed us to evaluate the performance of all the examined DCNN architectures in a reasonable amount of time.

Once we identified the most promising CADe system using the 90/10 split, we perform a 5-fold cross-validation to increase the statistical significance of our results using a 80/20 ratio to randomly split the data. This leads to a training set of size 815, and a validation set of size 203, out of the 1018 images of the dataset.

During training, the sub-volumes are randomly extracted from the CT images of the training set, and are normalized according to an estimate of the normal distribution of the voxel values in the dataset. A sub-volume is considered to be a nodule instance only if at least one of its slices contains, or is fully contained in, a complete 2D region of interest of a nodule. Alternatively, a sub-volume is considered to be a non-nodule instance only if none of its voxels is part of a nodule. The sampling from both the training and validation sets is done in a balanced fashion. This means that, on average, there is a similar number of non-nodule and nodule instances being extracted from these sets, and consequently, that our DCNN processes a similar number of non-nodule and nodule instances during training.

The goal of the backpropagation algorithm is to find a mapping between sub-volumes of CT images and the probability of them containing lung nodules. In other words, the backpropagation algorithm is expected to adjust the parameters of the DCNN so that given a previously unprocessed sub-volume of a CT scan, the network will output the value 1 if the sub-volume is considered to be a nodule instance, and the value 0 if the sub-volume is

considered to be a non-nodule instance. Similarly, given a batch of multiple sub-volumes of CT scans, the network is expected to output a list of probabilities indicating whether each of the sub-volumes is considered a nodule instance or not.

Training ends after 40 epochs of 5000 batches each, where the size of each batch is 128. The batch size represents the number of input instances that are being processed by the DCNN in each iteration of the backpropagation algorithm. Having a batch size of 1 leads to a pure stochastic gradient descent algorithm since the weights of the neural network will be updated after each processing of a single training instance. Alternatively, a batch size that is greater than 1, but is less than the number of all possible sub-volumes in the training set, leads to a mini-batch gradient descent algorithm.

The word epoch is commonly defined as the time it takes to process all the examples in the training set, which in our case equals to the number of all possible sub-volumes of CT scans in the training set. This number is very large, so we define an epoch as simply the time it takes to process 5000 batches, where each batch contains 128 sub-volumes of CT scans. As mentioned above, these sub-volumes are randomly extracted from the CT images of the training set during training.

Finally, we use a fixed learning rate of 0.001. Higher learning rates were examined but these led to significant fluctuations in the loss function. This is expected since using a higher learning rate means that the weights of the network are adjusted more rapidly at each iteration of the backpropagation algorithm (Chapter 2.4). As the learning rate increases, traversing the search space, i.e. traversing the space of all possible network weights, is becoming more similar to a random search algorithm. Lower learning rates have also been examined, and while these resulted in a steady decrease of the loss function, the pace by which the decrease occurred was lower.

There are two issues that commonly arise when training Deep Neural Networks (DNNs). These are overfitting and computation time. Overfitting occurs when a statistical model

captures patterns and regularities that are present in the training set but are not found outside of it. As a result, once such a model is applied to a previously unprocessed instance of the problem, it would not perform as well as it would on an instance from the training set. It generally happens when a model is excessively complex, such as having too many parameters relative to the number of instances in the training set.

Training DNNs is computationally expensive, and one has to take this into consideration when deciding which size of network to use and which hardware to run it on. We executed our experiments on the Graphics Processing Unit (GPU), either an Nvidia Tesla K80 or a GeForce GTX 960 GPU, which significantly improved the running-time of our system compared to only utilizing the CPU. To make this possible, we used Torch [CBM02], a scientific computing framework with wide support for machine learning algorithms, and Nvidia's cuDNN [Che+14b], a GPU-accelerated library of primitives for DNNs.

4.3 Predicting the location and boundary of lung nodules

Once the training of a CNN is complete, and given a previously unprocessed three dimensional CT image of size $D\times512\times512$ ($D\in[65, 764]$), we apply the CNN multiple times throughout the CT image using a fixed size sliding window, and compute a three dimensional voting grid of size $D\times512\times512$ by averaging the outputs of the CNN in the various sliding window positions. More specifically, the receptive field of the CNN is moved throughout the CT image in an ordered fashion so that it is applied to all the voxels in the image. For example, when the receptive field size is set to $5\times20\times20$, the receptive field is moved 10 voxels at a time for the in-plane axes, and 1 voxel at a time for the depth axis. Whenever a CNN is applied, its output value is added to all the entries in the voting grid that correspond to its receptive field. Then, each entry in the voting grid is divided by the number of times a CNN was applied to its corresponding voxel.

This procedure is illustrated in Figure 1.2 which shows a high-level diagram of DeepCADe. It illustrates how a single network is applied to two overlapping receptive fields (sub-volumes of a CT scan), and how the output of the network is aggregated into a single voting grid, which describes the predicted probability of a nodule in each location in the original scan.

Each entry in the resulting voting grid provides us with an estimate, in the range $[0, 1]$, to whether its corresponding voxel is part of a lung nodule. We use this voting grid, together with two thresholds (Threshold A and Threshold B), to predict the location and boundary of lung nodules. We do so by considering each adjacent set of entries, which have values greater than Threshold B, and have at least one value greater than Threshold A, to represent a predicted lung nodule.

Having a voting grid contributes to our system's ability to detect nodules with complex shapes. More specifically, decreasing the stride value in the voting phase of our algorithm will lead to an increase in the number of times the network, with its respective receptive fields, is applied for each of the entries in the voting grid. Consequently, it is expected that there would be more variation among the values of the voting grid, and therefore the nodule predictions, which are determined with Threshold A and Threshold B, are more likely to be complex in shape.

This capacity might be diminished in other related work [Ric+11; Gol+09; MHR10; Tan+11], where a spherical shape of lung nodules is usually assumed. This can cause the system to ignore nodules of peculiar shapes such as nodules that contain a non-nodule region within them or nodules that have a flat looking shape.

Also, this design of having a voting grid allows us to average the outputs of multiple classifiers in a fairly straightforward way. More specifically, the same way we compute a three dimensional voting grid of size $D \times 512 \times 512$ by averaging the outputs of a single CNN in the various sliding window positions, we can also use a number of different CNNs, with possibly different receptive field sizes, to compute the same three dimensional grid. Then,

each entry in the voting grid would be divided by the number of times any of the CNNs were applied to its corresponding voxel. This can be viewed as a bootstrap aggregating (bagging) algorithm which, in addition to a few other ensemble methods, is discussed in Chapter 2.5.

4.3.1 Threshold B

As mentioned above, each entry in the resulting voting grid provides us with an estimate, in the range $[0, 1]$, to whether its corresponding voxel is part of a lung nodule. We use this voting grid, together with two thresholds (Threshold A and Threshold B), to predict the location and boundary of lung nodules. We do so by considering each adjacent set of entries, which have values greater than Threshold B, and have at least one value greater than Threshold A, to represent a predicted lung nodule.

In Section 5.6, we explore two ways to control the size of nodule predictions our system makes, i.e. by examining different values of Threshold B or ignoring nodule predictions that are larger than a predefined size. Decreasing the value of Threshold B leads to an increased size of nodule predictions made by our CADe system. This needs to be carefully controlled and limited so that our CADe system does not produce nodule predictions that are too big. Such nodule predictions will not be of much help to radiologists since one of the primary goals of any CADe system is to provide its users with a compact approximation to where lung nodules might be present. Failure to do so can undermine the radiologists' trust in the system.

Consequently, we assume that the value of Threshold B needs to be chosen so that it results in an average size of nodule predictions that is similar to the average size of lung nodules in the dataset. In Section 5.6 we discuss the dynamics between the value of Threshold B and the nodule prediction size, and examine whether using Threshold B alone is sufficient to control the size of the nodule predictions.

The optimal value for Threshold B is not known and is likely to vary across CT scans due to a number of reasons. These include the scanner and scanner configuration used to generate

the scan, and the physiology and clinical state of the person being scanned. To overcome this uncertainty, DeepCADe can be deployed in a semi-automated fashion by allowing the user of the system to adjust the value of Threshold B in real time and see the results for himself. The discussion in Section 5.6 explores how the predictions of the system are affected by such adjustments, and consequently, gives us a better understanding of how the user experience of such a semi-automated CADe system might be.

4.3.2 Threshold A

Threshold A is the value that is used to control the sensitivity of our CADe system, and therefore is used to compute the FROC curve of the system. This is elaborated in Section 4.4. The FROC curve allows us to better understand the performance of the CADe system, and the FROC of the 5-fold cross validation experiment provides us a good estimate to how well the system will perform in real time.

When deploying the CADe system, e.g. during a radiologist's interpretation of a previously unprocessed CT image, Threshold A needs to be determined. There are two ways by which this can be achieved, each having its pros and cons:

(1) Threshold A can be fixed to a pre-defined value based on the tradeoff between the average sensitivity and false positive per scan values of the FROC curve. The disadvantage of this approach is that the system is expected to produce a varying number of nodule predictions, potentially false positives, for different unprocessed CT scans, which has the potential to undermine the radiologist's trust in the system. This is expected to happen since the voting values of different CT images tend to vary, and having a fixed Threshold A value for all images holds the risk of predicting too many nodules for one scan, and not predicting enough for another. The advantage of this approach is that it is easy and fast to compute.

(2) Threshold A can be computed dynamically for each CT scan to achieve a pre-defined number of predictions. More specifically, before anything is presented to the user, the system

will examine a range of values for Threshold A until finding one that results in a predefined number of nodule predictions. As the value of Threshold A decreases, more nodule predictions might be added to the output of the system. This approach ensures that the user will always be presented with the same number of predictions at the start of his/her interpretation process, and then, depending on whether the predictions turned out to be actual nodules or not, the user can ask the system to present him/her with more nodule predictions. This can be done by manually lowering the value of Threshold A. A disadvantage of this approach is that it is more computationally expensive.

4.3.3 Using both Threshold A and Threshold B

Figure 4.3 illustrates how Threshold A and Threshold B are applied to a one dimensional sequence of voting grid values to detect regions which contain lung nodules. As mentioned above, each entry in the voting grid is an estimate, in the range $[0, 1]$, to whether its corresponding voxel in the CT scan is part of a lung nodule. We consider each adjacent set of voting grid entries, which have values greater than Threshold B, and have at least one value greater than Threshold A, to represent a predicted lung nodule.

In Figure 4.3, we examine a hypothetical one dimensional sequence of voting grid values and see how changes in Threshold A and Threshold B affect the nodule predictions that are computed from it. As expected, Figures 4.3(A-B) show that a decrease in the value of Threshold B can lead to an increase in the size of nodule predictions, and Figures 4.3(B-C) show that a decrease in Threshold A can lead to an increase in the number of nodules being detected.

4.4 Performance Evaluation using a FROC Curve

A Receiver Operating Characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The

ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true positive rate is also known as sensitivity, recall or probability of detection, and the false positive rate is also known as the fall-out or probability of false alarm. More formally, given the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values of the binary classifier, the true positive rate (TPR) is defined as follows:

$$TPR = \frac{TP}{TP + FN} \tag{4.1}$$

and the false positive rate (FPR) is defined as follows:

$$FPR = \frac{FP}{FP + TN} \tag{4.2}$$

ROC analysis allows the user to select possibly optimal classifiers, i.e. classifiers which have a TPR of 1 and a FPR of 0, and to discard suboptimal ones independently from the class distribution. ROC analysis is related in a direct and natural way to cost/benefit analysis of diagnostic decision making, and therefore it has been commonly used in areas such as medicine and radiology, and is increasingly used in machine learning research.

Similarly, a Free-response Receiver Operating Characteristic (FROC) curve is a tool for characterizing the performance of a free-response system at all decision thresholds simultaneously [Cha13]. A CADe system is considered to be a free-response system since its aim is not just to predict whether a medical image contains an abnormality or not (i.e. it is not a binary classifier), but to predict the exact location and boundary of an undefined number of abnormalities in the image. If the former was true, a conventional ROC curve would suffice to characterize the performance of the system. The FROC curve was first introduced in [Mil69], where it was used to visualize the performance of a free-response task from the auditory domain. Its importance for radiology applications was first recognized by [Bun+77], and ever since it has been widely used to characterize the performance of CADe systems and other localization tasks.

Given Threshold A (the discrimination threshold in DeepCADe), we define the FP value to be the number of nodule predictions, made by DeepCADe, that do not contain any voxel of an annotated lung nodule. Also, we define the TP value to be the number of annotated lung nodules that have been successfully detected by DeepCADe, and the FN value to be the number of annotated lung nodules that have not been detected by DeepCADe. Given the TP and FN values, the sensitivity (TPR) of DeepCADe can be computed using Equation 5.1.

A single FROC curve is plotted by computing, for a range of Threshold A values, the sensitivity and FPs per scan values, and using linear interpolation to connect between the computed points. More specifically, for each CT scan, we initialize the value of Threshold A with the maximal value in the corresponding voting grid. Then, after computing the sensitivity and FP values for this specific value of Threshold A, we decrease Threshold A by 0.01 which usually leads to more FPs being produced by the system. This process continues until the number of FPs that the system produces is greater than 20. Finally, for each FP per scan value, the sensitivity values are averaged among all the CT images in the validation set.

The total number of CT images in the validation set assuming a 90/10 split of the LIDC-IDRI dataset is 102, all of which are used to validate the performance of a single CNN during its training on sub-volumes of CT scans. However, some of these images are ignored when computing the FROC curve of the entire CADe system. This is because we can only consider CT images that contain at least one lung nodule when performing a FROC analysis. CT images that contain no nodules are ignored since they will always have a sensitivity of zero, regardless of the number of FPs the system produces on them. Table 4.2 describes, for agreement level 1 to 4, the effective number of CT images in the validation set and the number of nodules they contain.

Table 4.2: The effective number of CT scans in the validation set, i.e. the number of scans which contain at least one lung nodule, and the total number of lung nodules in the validation set, for agreement level 1 to 4.

Agreement level	Effective size of validation set	Number of lung nodules in the validation set
1	88	251
2	84	190
3	73	134
4	57	86

Alternatively, the average number of CT images in the validation set assuming a 80/20 split of the LIDC-IDRI dataset is 203, and the average number of CT images which contain at least one nodule at agreement level 3 is 140. These values are averaged over five complementary splits of the LIDC-IDRI dataset as part of the final 5-fold cross validation analysis.

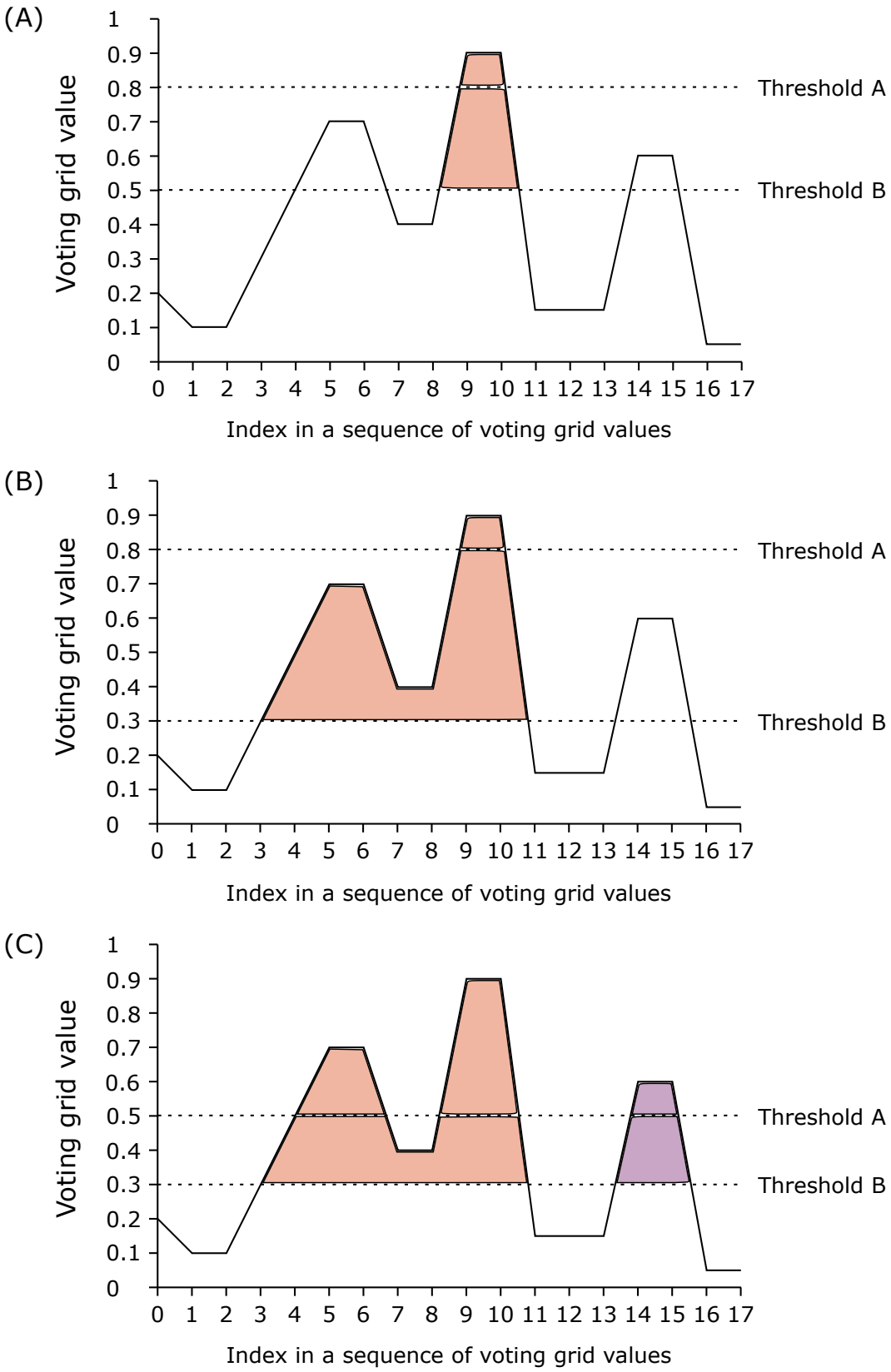

Figure 4.3: An illustration of how Threshold A and Threshold B are applied to a one dimensional sequence of voting grid values to detect regions which contain lung nodules.

Chapter 5

Experimental Results

In this chapter, we perform a systematic study on the application of Deep Convolutional Neural Networks (DCNNs) for the detection of lung nodules in thoracic CT scans. We explore some of the meta-parameters that affect the performance of such models, which include:

1. the depth of the classifier module of the DCNN, i.e. the number of fully-connected layers the classifier is composed of,

2. the size of the classifier module of the DCNN, i.e. the number of learnable parameters the classifier is composed of,

3. the depth and size of the feature extraction module of the DCNN, i.e. the effect convolutional layers have on the performance of the CADe system,

4. the benefit of using a ReLU activation function compared to a sigmoid and tanh functions,

5. the receptive field of the network, which defines the dimensions of its input, i.e. how much of the CT scan is processed by the network in a single forward pass,

6. two ways to control the nodule prediction size, i.e. by using a dedicated threshold value or ignoring nodule predictions that are larger than a predefined size, and

7. the four agreement levels, which define four sets of nodule annotations according to the level of agreement among four experienced radiologists.

The goal of each of the above experiments is to examine the effect of a single meta-parameter on performance, after which that meta-parameter remains fixed for all consecutive experiments. This way of exploring the meta-parameter space of the system significantly reduces the number of experiments that needs to be conducted compared to exploring all possible combinations of the examined meta-parameter values. Consequently, since training and evaluating the performance of a single network is computationally expensive, this approach makes it practical to examine the wide range of meta-parameter values that are discussed here. However, this approach carries the risk of missing out a successful combination of meta-parameter values. For example, since we are exploring the size and depth of the network first, and the receptive field of the network second, it is possible that smaller networks would yield better results after an increase in the size of the receptive field.

We use a Free-response Receiver Operating Characteristic (FROC) curve, as discussed in Chapter 4.4, to evaluate the various configurations of our CADe system codenamed DeepCADe, and compare our most promising configuration to some previous work where the LIDC-IDRI dataset has also been used for validation. The FROC curve illustrates the performance of a free-response system at all decision thresholds simultaneously by plotting its sensitivity (true positive rate) and FPs per scan values for a range of decision thresholds. The sensitivity of a free-response system is defined as follows:

$$sensitivity = \frac{TP}{TP + FN} \tag{5.1}$$

Furthermore, we examine a number of CT slice images to illustrate the overlap between the nodule predictions made by DeepCADe and the actual annotated lung nodules. Finally, we perform a 5-fold cross-validation on the most promising configuration of DeepCADe to increase its statistical significance.

Appendix A contains a complete description of all the networks discussed in this chapter.

Table 5.1: The number of fully-connected hidden layers the classifier module is composed of, the number of units per layer, the sensitivity at 10 FPs per scan, and the average nodule prediction size, for experiments $e1$, $e2$, $e3$, and $e4$. The total number of learnable parameters in all four experiments is the same.

Experiment	Number of hidden layers	Number of units per layer	Sensitivity at 10 FPs per scan	Average nodule prediction size (in voxels)
$e1$	1	9850	63.72%	19,349
$e2$	2	3588	63.62%	8,497
$e3$	3	2709	80.59%	62,702
$e4$	4	2274	68.08%	17,396

5.1 Classifier Depth

As described in Chapter 4.2, our DCNN is composed of two modules, i.e. feature extraction and classification. The first module of the DCNN is composed of multiple volumetric convolution, rectified linear units (ReLU), and max pooling layers, and the second module is the classifier which is composed of multiple fully connected layers and non-linearity operations. Here, we examine the effect that the depth of the classifier, i.e. the number of fully-connected layers the classifier is composed of, has on the performance of DeepCADe. To examine this, we conducted 4 experiments ($e1$, $e2$, $e3$, $e4$) in which the classifier module is composed of 1, 2, 3, and 4 fully-connected hidden layers, respectively. We adjusted the number of units in each hidden layer so that the total number of learnable parameters in all experiments is the same. Table 5.1 describes the number of hidden layers the classifier module is composed of, the number of units per layer, the sensitivity at 10 FPs per scan, and the average nodule prediction size, for each of these four experiments.

Figure 5.1 suggests that the ideal number of fully-connected layers in the classifier is 3, since the area under the curve for the classifier with 3 hidden layers is the largest. Adding more layers to the classifier does not necessarily correlate with better performance. In addition to looking at the area under the curve for the entire range of FP per scan values (0 to 20), we can also examine smaller ranges of FP per scan values, each of which has a specific meaning and implications.

For example, it is reasonable to say that an end user of the system can tolerate about 4 to 6 false positives per scan and examine them manually. This is why it is common to report the sensitivity values around this range [Ric+11; Gol+09; MHR10; Tan+11]. Having more FPs per scan might undermine the user's trust in the system, while having too little FPs per scan can lead to a significant drop in sensitivity. Therefore, the area under the curve for this range of values (between 4 to 6 FPs per scan) provides us with an estimate of how well the system can perform in "reasonable" settings.

Alternatively, examining the range of 10 to 20 FPs per scan values shows us how well the system can perform if the number of FPs per scan is not a major concern. This can be the case in certain user interfaces where the user can easily increase/decrease the number of FPs per scan and see the effect it has on the nodule predictions of the system in real-time throughout the entire scan.

As discussed in Chapter 4.4, the FROC curve illustrates the performance of a free-response system at all decision thresholds simultaneously, but note that the number of decision thresholds that are used to plot each FROC curve varies between one experiment to another and depends on the system in examination and the scheme by which the decision thresholds are updated.

In addition to sensitivity, having 3 hidden layers in the classifier module of our DCNN results in an average nodule prediction size of 62,702 voxels, while the actual average nodule size is 6056 voxels. This means that the average nodule prediction size for experiment $e3$ is roughly 10 times larger than the actual average nodule size. The predicted versus real nodule prediction size issue is discussed in more detail in Section 5.6 which examines a number of remedies for this issue. Having a nodule prediction size that is too large can be an issue since the user will then need to examine the predicted area and identify the area of interest manually, while having a nodule prediction size that is too small can be an issue since the user might overlook it. This again depends on the way by which the output of DeepCADe

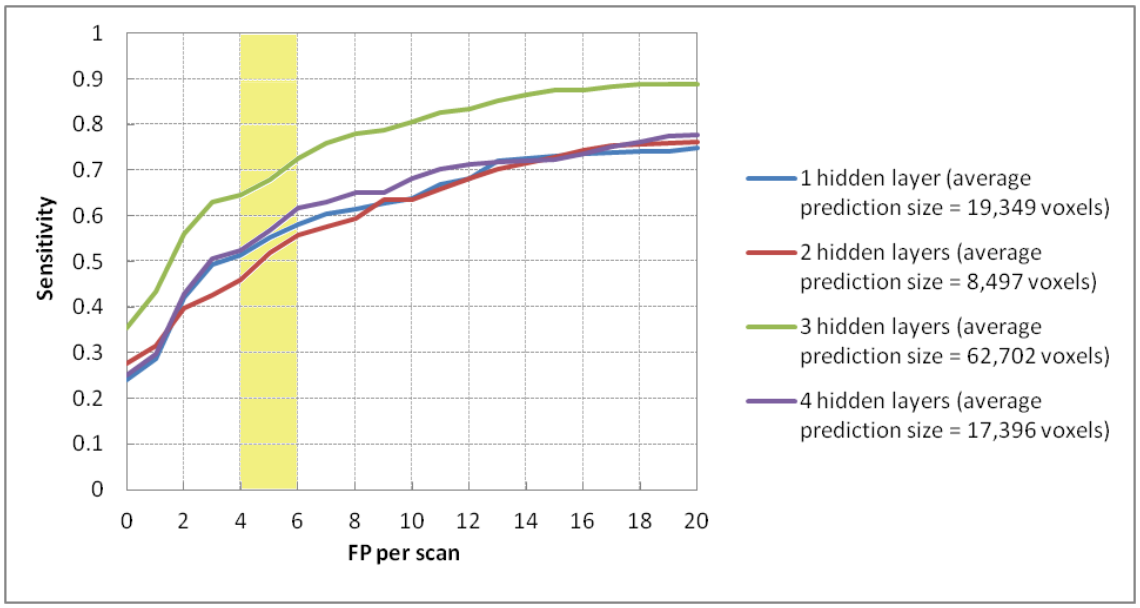

Figure 5.1: FROC curve of 4 experiments ($e1$, $e2$, $e3$, $e4$) in which the classifier is composed of 1, 2, 3, and 4 fully-connected hidden layers, respectively.

is presented to the user, and the degree by which the user can interact with the system.

5.2 Classifier Size

Here, we examine how changing the number of units in each of the hidden layers of the classifier affects its performance. More specifically, we conduct 4 experiments ($e3\_500$, $e3\_1000$, $e3\_1500$, $e3\_2709 = e3$) in which the number of units in each hidden layer is 500, 1000, 1500, and 2709, respectively. Notice that here the number of hidden layers is the same for all experiments, i.e. 3 hidden layers, but the size of each layer varies. This means that the total number of learnable parameters is not fixed, and is increasing with the number of units in each hidden layer.

Figure 5.2 shows that as the number of units in each fully-connected hidden layer increases, so does the performance of the CADe system. Another interesting observation is that a larger average nodule prediction size does not always translate into better performance. In $e3\_1000$, the average nodule prediction size is 100,622 voxels but the FROC is worse than the one for $e3\_2709$ where the average nodule prediction size is 62,702 voxels.

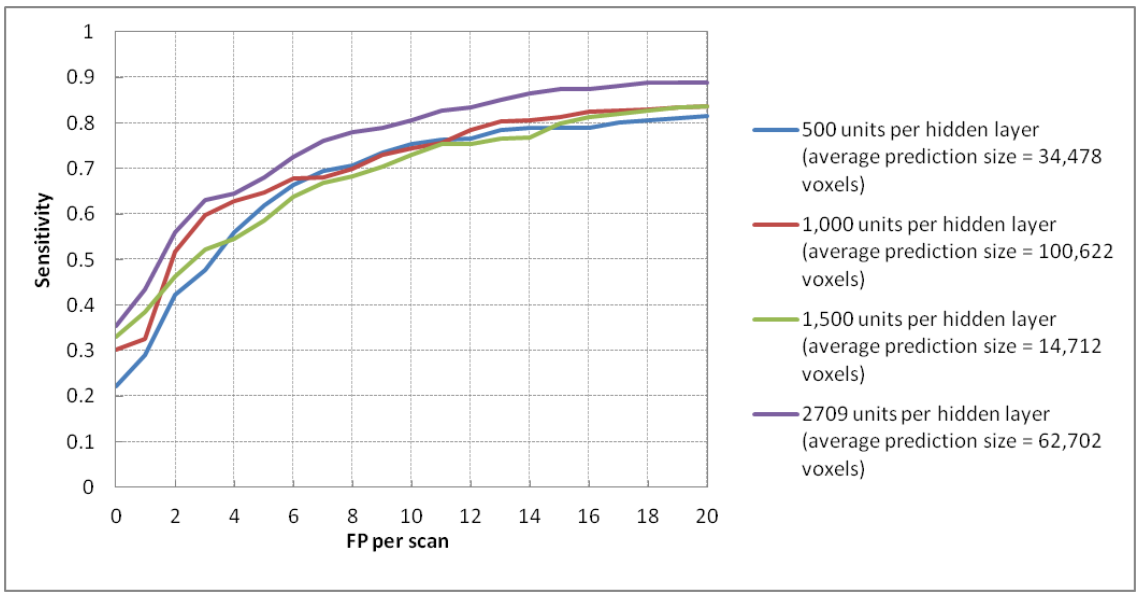

Figure 5.2: FROC curve of 4 experiments ($e3\_500$, $e3\_1000$, $e3\_1500$, $e3\_2709 = e3$) in which the number of units in each hidden layer is 500, 1000, 1500, and 2709, respectively.

Also, Figure 5.2 shows that, in the range of 2 to 6 FPs per scan, the sensitivity values for $e3\_1000$ and $e3\_2709$ are relatively close to each other. But for FP per scan values that are greater than 6, $e3\_2709$ shows better sensitivity compared to all other experiments. This means that it is sometimes necessary to increase the number of false positives so that the performance gain of one experiment becomes apparent compared to others.

In experiment $e3\_2709$, the total number of learnable parameters that the network, or more specifically the classifier module of the network, is composed of is the highest. As the size of the network increases, its representational power increases as well. This increase in representational power could be one of the reasons why $e3\_2709$ is performing better compared to the other experiments; it might allow the network to make better use of the feature maps that were learned during the feature extraction module of the network. Note that the feature extraction module in all of the above experiments ($e3\_500$, $e3\_1000$, $e3\_1500$, $e3\_2709 = e3$) is the same, so any increase in performance can only be due to the changes made in the classifier module of the network.

5.3 Feature Extractor Depth and Size

Here, we examine how changing the depth and size of the feature extraction module of the DCNN affects its performance. More specifically, we conduct 4 experiments ($e3\_0Conv$, $e3\_2Conv$, $e3\_4Conv = e3$, $e3\_6Conv$), in which the number of convolution layers is 0, 2, 4, and 6, respectively. Figure 5.3 shows the FROC of these experiments.

Similarly to the FROC curve of experiments $e3\_1000$ and $e3\_2709$ in Figure 5.2, Figure 5.3 also shows a point in the FROC curve of experiments $e3\_2Conv$ and $e3\_4Conv$, where the gap between the two becomes more significant. This branching occurs at around 2 false positive per scan, after which the performance of $e3\_4Conv$ outperforms that of $e3\_2Conv$.

The addition of convolution layers has been proposed to improve the performance of DCNNs due to two main reasons [BCV13]. First, deep architectures promote the re-use of features, which means that the number of paths from input to output, i.e. the number of ways to re-use the features learned by the network, grows exponentially with its depth. Second, deep architectures can potentially lead to progressively more abstract features at higher layers of representations. It is possible that the addition of 2 convolution layers to $e3\_2Conv$ is one of the reasons for the increase in performance of $e3\_4Conv$ compared to $e3\_2Conv$.

However, $e3\_6Conv$, which is composed of 2 additional convolution layers, shows reduced performance compared to $e3\_4Conv$ regardless of the number of FPs per scan. This means that the addition of convolution layers to the feature extraction module of the DCNN does not always correlate with better performance on the validation set. In other words, it means that the increase in the number of learnable parameters, and therefore the increase in representational power, does not necessarily leads to better performance on the validation set. This can be a sign of overffiting, where the neural network has learned patterns and regularities that are found in the training data but that are not related to the task at hand.

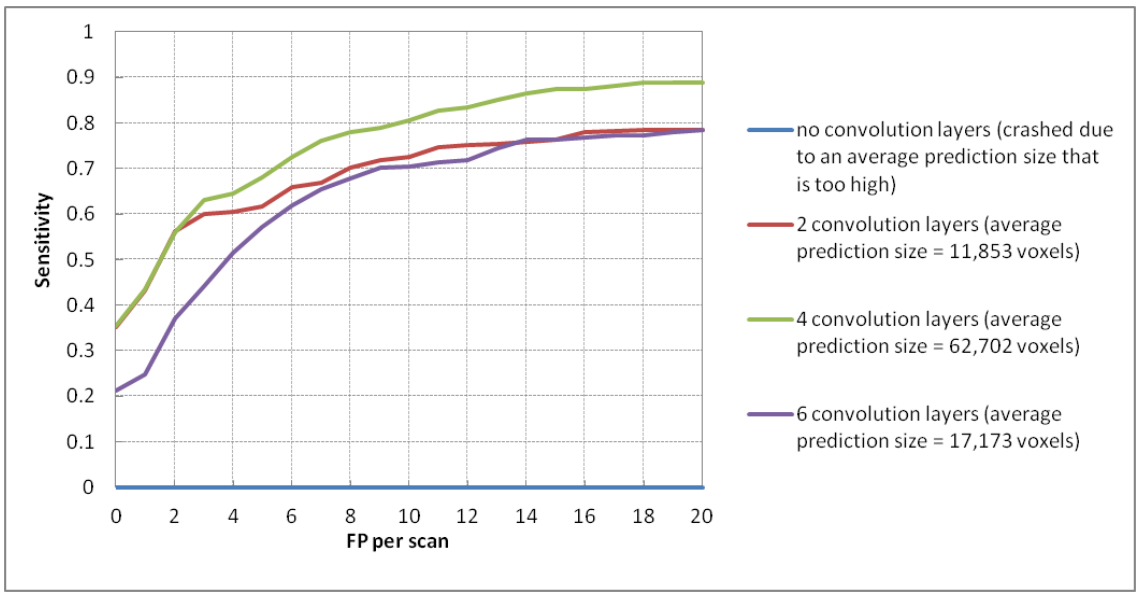

Figure 5.3: FROC curve of 4 experiments ($e3\_0Conv$, $e3\_2Conv$, $e3\_4Conv = e3$, $e3\_6Conv$) in which the number of convolution layers is 0, 2, 4, and 6, respectively.

In experiment $e3\_0Conv$, the feature extraction module of the DCNN is removed, i.e. no convolution layers are used. Consequently, the classifier is being fed with the raw CT image values that are in its receptive field. $e3\_0Conv$ produces nodule predictions that are very large, which causes the system to crash due to an out of memory error. More specifically, since the nodule predictions it produces are so large, there is not enough memory to store their voxel indices, which is required to determine whether it hits an actual nodule or not.

5.4 Activation Functions

Here, we examine how using different activation functions affects the performance of the CADe system. More specifically, we conduct 3 experiments ($e3\_sig$, $e3\_tanh$, $e3\_relu = e3$) in which the activation function being used is Sigmoid, Hyperbolic Tangent, and ReLU, respectively. These activation functions are applied after every convolution and fully-connected layer on each of their constituting nodes. Figure 5.4 shows the results of these experiments.

While the FROC curve of $e3\_tanh$ is comparable to $e3\_relu$, the average nodule prediction size of $e3\_tanh$ is significantly larger compared to the one produced by $e3\_relu$. This makes

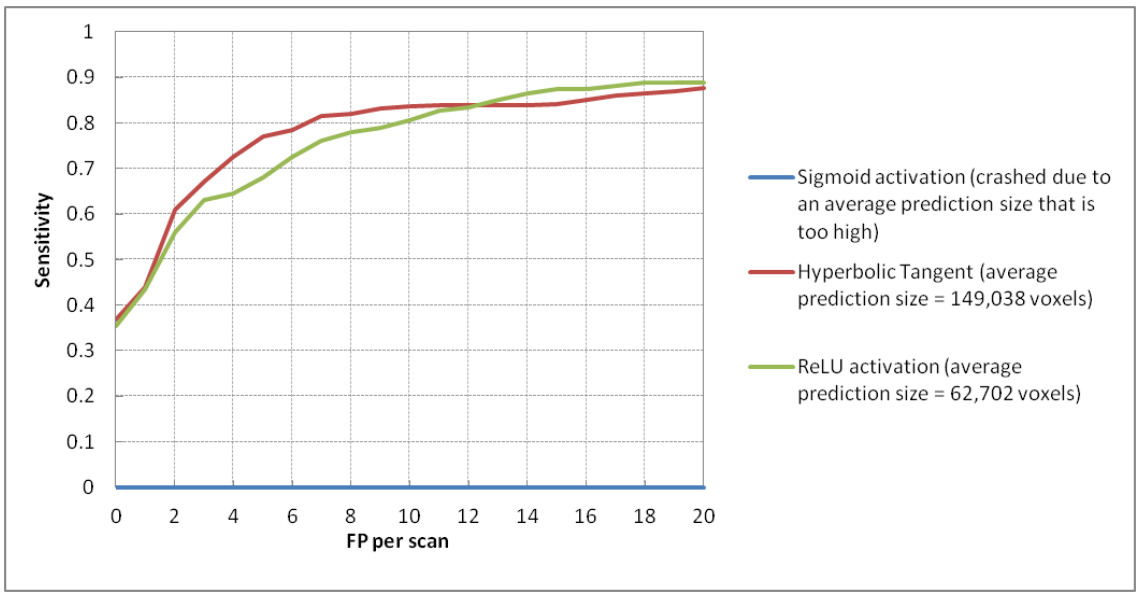

Figure 5.4: FROC curve of 3 experiments ($e3\_sig$, $e3\_tanh$, $e3\_relu = e3$) in which the activation function being used is Sigmoid, Hyperbolic Tangent, and ReLU, respectively.

the use of the rectified linear unit activation function preferable to the hyperbolic tangent function. Also, similarly to $e3\_0Conv$, the system has crashed due to an out of memory error when applying the sigmoid function as an activation function ($e3\_sig$). This again is due to an average nodule prediction size that is too large.

The advantage of using non-saturated activation functions such as ReLU has been suggested to be due to their capacity to solve the so called vanishing gradient problem and to accelerate the convergence speed of the learning algorithm [Xu+15]. The vanishing gradient problem can occur when training certain neural networks with gradient based methods such as backpropagation. As discussed in Chapter 2.4, gradient based methods learn a parameter's value by understanding how a small change in the parameter's value will affect the network's output. If a change in the parameter's value causes very small change in the network's output, the network simply can not learn the parameter effectively, which is a problem.

Saturated activation functions such as sigmoid or hyperbolic tangent map their input into a very small output range in a very non-linear fashion. As a result, there are large regions of

the input space which are mapped to an extremely small range. In these regions of the input space, even a large change in the input will produce a small change in the output, hence the gradient is small. This gets even worse as the number of layers in the architecture increases. In contrast, non-saturated activation functions such as ReLU do not have this property and therefore are better suited for training deep neural networks.

5.5 Receptive field

Implementing a neural network which processes an entire CT scan in a single forward pass is impractical given the computational resources we have available at our disposal. Having a significantly smaller receptive field and taking a sliding window approach is one way to overcome this limitation. We examine how the size of the receptive field affects the performance of the CADe system. The receptive field of the network defines the 3D region of the CT scan which is being processed by the network. Larger receptive fields allow for more context to be processed by the network, so for example proximity of nodule predictions to more distant organs can be taken into consideration by networks which have larger receptive fields.

As it turns out, the receptive field has a significant effect on the performance of DeepCADe. Figure 5.5 shows 5 experiments ($e3\_r20 = e3$, $e3\_r40$, $e3\_r60$, $e3\_r80$, $e3\_r60.10$) in which the receptive field dimensions are $20 \times 20 \times 5$, $40 \times 40 \times 5$, $60 \times 60 \times 5$, $80 \times 80 \times 5$, and $60 \times 60 \times 10$, respectively. It shows that using a receptive field of size $60 \times 60 \times 5$ leads to an outstanding improvement in sensitivity compared to using a receptive field of $20 \times 20 \times 5$. More specifically, assuming the system is configured to allow 5 false positives (FPs) per scan, $e3\_r60$ produces an absolute 25% improvement in sensitivity compared to $e3\_r20$. Alternatively, assuming 10 or 20 false positives per scan, $e3\_r60$ produces an absolute 15.6% or 9% improvement in sensitivity compared to $e3\_r20$.

We hypothesize this positive effect on results to be due to the network's capacity to process a larger region of the CT scan at a time, and therefore, have a larger context to

work with. For example, a larger receptive field may allow the network to consider relative distances of the nodules it predicts to a larger number of region of interests such as the wall of the lungs, blood vessels, or the heart. Additionally, a larger receptive field might compensate for the fact that DeepCADe does not include any additional lung segmentation procedure, and therefore, having a larger context allows it to explicitly determine the location of the nodule predictions in relation to the lung walls.

While this hypothesis explains the improved performance of $e3\_r60$ compared to $e3\_r20$ and $e3\_r40$, it does not explain why we see a decrease in performance with $e3\_r80$ and $e3\_r60.10$ which have a larger receptive field compared to $e3\_r60$. Having a larger receptive field, i.e. increasing the dimensionality of the input, increases the likelihood of overfitting and therefore can lead to this observed decrease in performance on the validation set. More specifically, having a larger receptive field increases the capacity of the classifier to find patterns and regularities that are found in the training set, but that are not useful for the general case as represented by the validation set.

Additionally, Figure 5.5 shows that $e3\_r60$ produces nodule predictions that are, on average, significantly smaller than the ones produced by $e3\_r40$ without any significant loss in sensitivity. More specifically, $e3\_r60$ produces an average nodule prediction size of 17,248 voxels which is about four times smaller than the average prediction size of $e3\_r40$, and is about three times larger than the average nodule size of 6,056 voxels. $e3\_r80$ has even better performance in this regard, and is able to produce an average nodule prediction of 3,365 voxels, which is nearly half of the average nodule size.

Having a CADe system that produces nodule predictions that are as compact as possible is crucial for its applicability in real world scenarios. Having nodule predictions that are too big will not be very helpful to radiologists since they will then have to examine the over-sized predicted regions and look for abnormalities themselves. The issue of prediction compactness is discussed in more detail in Section 5.6 where we examine two approaches by which nodule

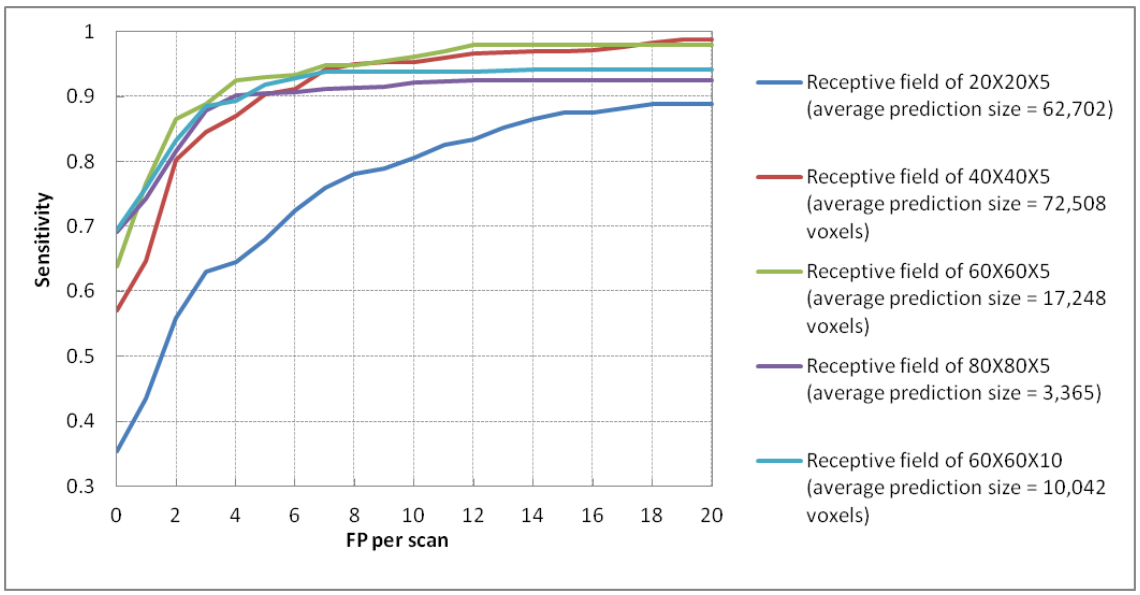

Figure 5.5: FROC curve of 5 experiments ($e3\_r20 = e3$, $e3\_r40$, $e3\_r60$, $e3\_r80$, $e3\_r60.10$) in which the receptive field dimensions is $20 \times 20 \times 5$, $40 \times 40 \times 5$, $60 \times 60 \times 5$, $80 \times 80 \times 5$, and $60 \times 60 \times 10$, respectively.

predictions can be controlled. In addition, Section 5.10 includes a number of CT slice images to illustrate the overlap between annotated lung nodules and nodule predictions.

The maximum/minimum/average in-plane diameter of a lung nodule in the dataset is 76/1/15.5 pixels, respectively, and the maximum/minimum/average number of slices a nodule is composed of is 56/1/6, respectively. So choosing the ideal receptive field is not clear, especially since no image rescaling has been performed. The experiments in this section make it more clear what the receptive field of the network should be, and support the idea of using a three dimensional receptive field. We hypothesize that using a three dimensional receptive field is successful due to the fact that lung nodules are three dimensional objects, and having a receptive field that captures multiple image slices at a time allows the DCNN to extract volumetric features that can be beneficial to the successful detection of lung nodules.

5.6 Controlling the nodule prediction size

Here we examine two ways by which the nodule prediction size can be controlled, namely using Threshold B and ignoring nodule predictions that are larger than a predefined size. More specifically, we conduct 4 experiments ($e3\_r60\_t0.5$, $e3\_r60\_t0.4$, $e3\_r60\_t0.3$, $e3\_r60\_t0.3\_limit$) in which the value of Threshold B is 0.5, 0.4, 0.3, and 0.3, respectively, and where the last experiment is the only one that ignores nodule predictions that are larger than 234,436 voxels (a size that is double the maximal annotated nodule size of 117,218 voxels).

Figure 5.6 shows the results of these experiments. It shows that as the value of Threshold B increases, the average size of nodule predictions decreases and so does the sensitivity of the system, especially in the range of 0 to 10 FPs per scan. This makes sense since as the value of Threshold B increases, the nodule predictions become smaller and are less likely to hit a voxel that belongs to an annotated lung nodule.

In addition, and somewhat surprisingly, having a limit on the nodule prediction size actually increases the average prediction size. This can be explained as follows. Since there is a limit on the number of voxels a nodule prediction can contain, an over-the-limit nodule prediction that would otherwise be considered a single nodule will now be considered as several smaller nodule predictions. Consequently, assuming this nodule prediction is a FP, the number of FPs can potentially increase faster than it would have without the limit, and as a result cause the FROC computation to stop earlier, missing nodule predictions that would otherwise be added to the average. These nodule predictions can be of small size, and therefore missing them would lead to an increase in the average nodule prediction size.

5.7 Agreement level

The agreement level of a lung nodule defines the number of radiologists who made a nodule annotation which overlaps it. Nodules at agreement level j, where $1 \leq j \leq 4$, include all those

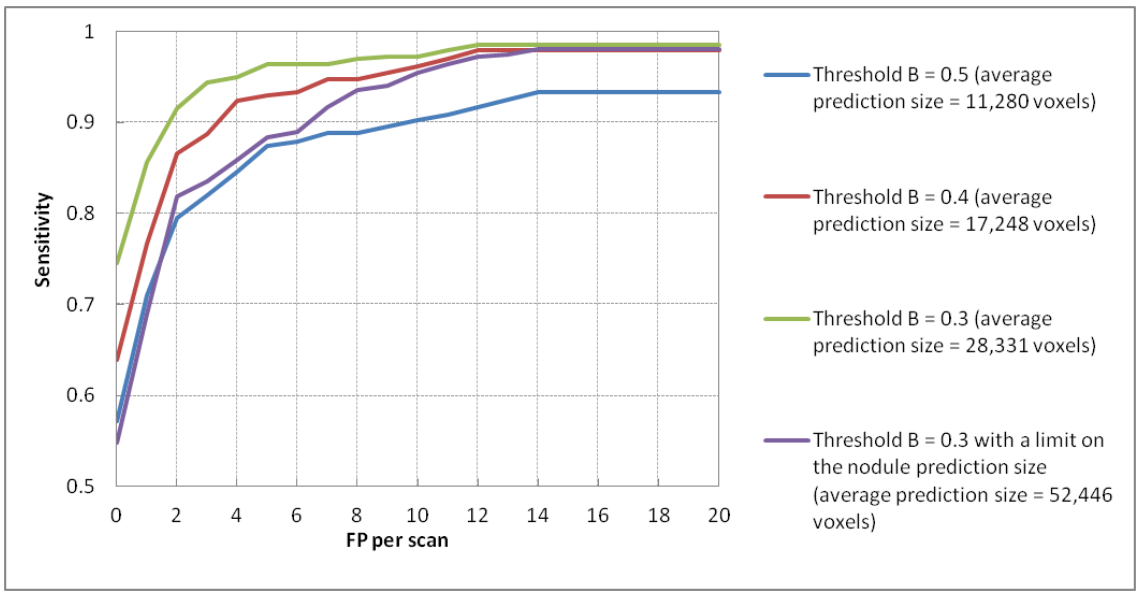

Figure 5.6: FROC curve of 4 experiments ($e3\_r60\_t0.5$, $e3\_r60\_t0.4$, $e3\_r60\_t0.3$, $e3\_r60\_t0.3\_limit$) in which the value of Threshold B is 0.5, 0.4, 0.3, and 0.3, respectively, and where the last experiment is the only one that ignores nodule predictions that are larger than 234,436 voxels.

nodules which were marked by at least j radiologists. We expect nodules at a higher agreement level to be more easily detectable by an automated system since these nodules were identified by more radiologists and therefore are assumed to have clearer boundaries.

Figure 5.7 illustrates the FROC curve of DeepCADe using $e3\_r60\_t0.3$ in four different configurations, one for each of the four agreement levels. Examining agreement level 1 to 3, Figure 5.7 shows that our system performs better on lung nodules that are at higher agreement levels, which is reasonable since these nodules are expected to be more easily detectable.

However, Figure 5.7 shows a decline in performance when considering only those lung nodules which have been annotated by all four radiologists. In this case, the classifier assumes that all nodules at agreement level 1 to 3 are not nodules at all. In other words, the classifier is required to distinguish between nodules at agreement level 4 and the rest of the nodules. This can be very challenging, especially when discerning between nodules at agreement level 4 and 3, since these might look very similar and can only be successfully classified by

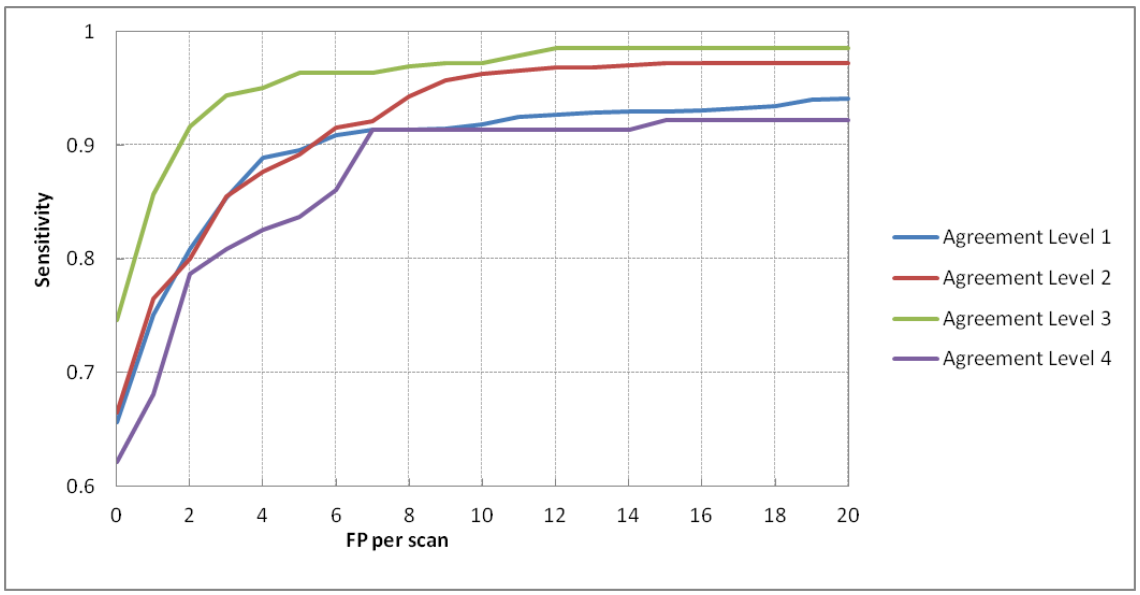

Figure 5.7: FROC curves of DeepCADe using $e3\_r60\_t0.3$ for nodules in the validation set at four agreement levels.

experienced radiologists. Consequently, there is a decline in performance when considering nodules at agreement level 4.

5.8 Cross-Validation

Cross-Validation is a model validation technique for assessing how accurately a predictive model will generalize to an independent data set [Koh+95]. One round of cross-validation involves partitioning a sample of data into two complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (called the validation set). In k-fold cross-validation, k rounds of cross-validation are performed using complementary partitions, and the validation results are averaged over the rounds. This means that each observation is used for validation exactly once.

Figure 5.8 shows the 5-fold cross validation FROC curves of DeepCADe using $e3\_r80$ with Threshold B of 0.4 and for nodules at agreement level 3. It shows that, on average, $e3\_r80$ achieves a sensitivity of 89.6% with 4 FPs per scan, or a sensitivity of 92.8% with 10 FPs per scan. Moreover, while the average size of nodules at agreement level 3 is 6,427 voxels,

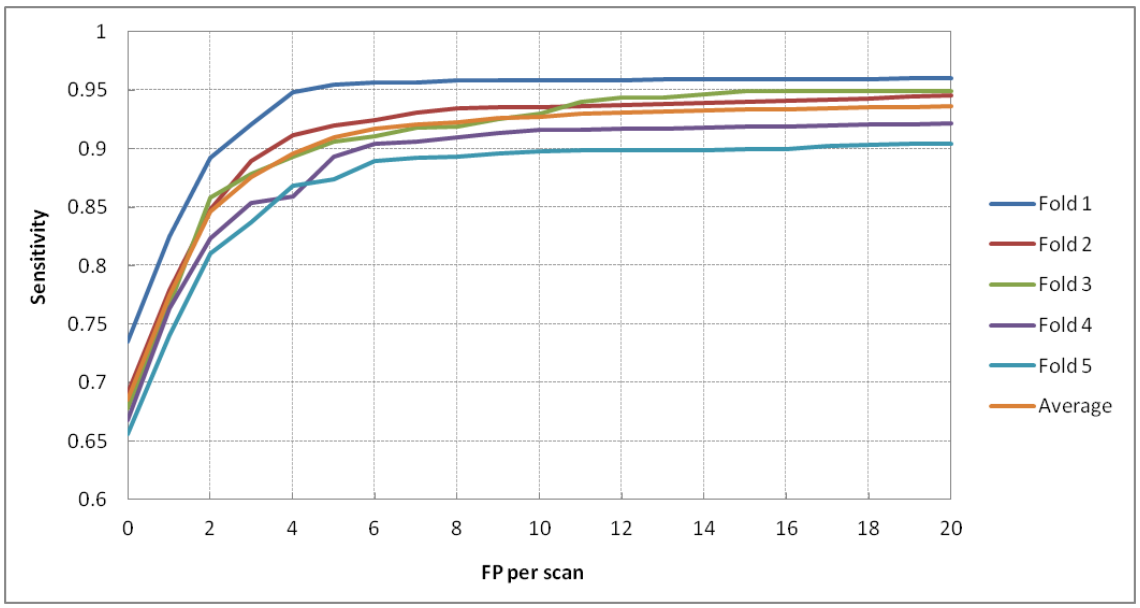

Figure 5.8: The 5-fold cross validation FROC curves of DeepCADe using $e3\_r80$ with Threshold B of 0.4 and for nodules at agreement level 3.

the average nodule prediction size, made by $e3\_r80$, is 30,113 voxels. This means that, on average, $e3\_r80$ produces nodule predictions that are 4.7 times larger than the average size of nodule annotations. This is not far from the ideal case where there is a perfect match between predicted and annotated lung nodules, and where any false positive is of minimal size.

Figure 5.8 also shows that the performance of the system is affected by the selection of the complementary training and validation sets. To better understand this variability in performance, Figure 5.9 illustrates the average 5-fold cross validation FROC curve with 1 standard deviation error bars. The standard deviation varies depending on the number of false positives per scan, but in general its value is roughly 3%. This provides us with an estimate to how much the results we report in this work are attributed to the selection of the complementary training and validation sets.

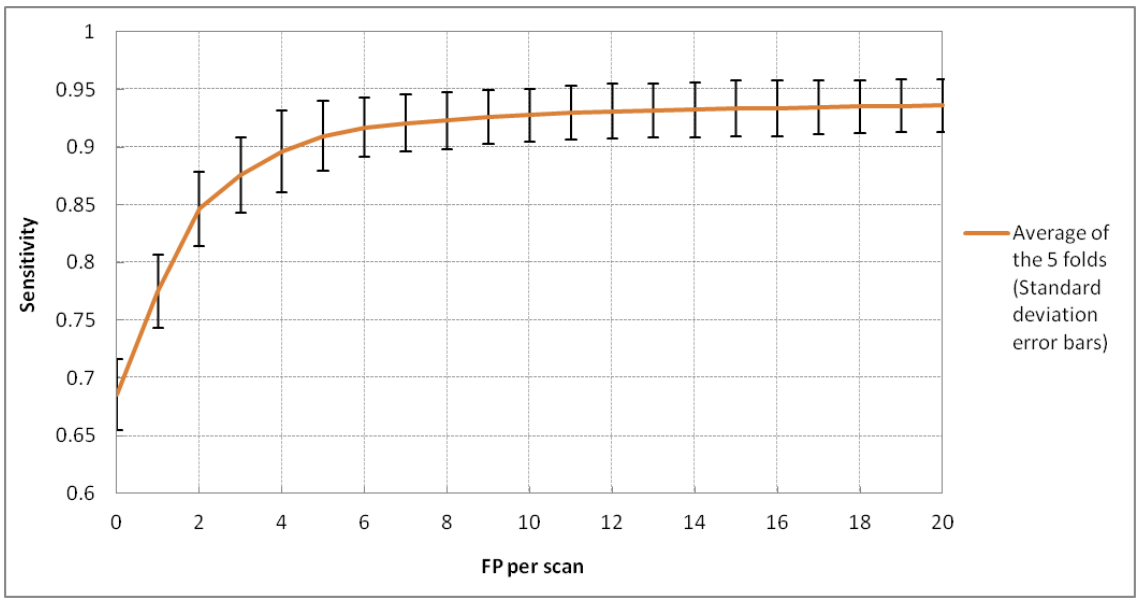

Figure 5.9: The average 5-fold cross validation FROC curve of DeepCADe using $e3\_r80$ with Threshold B of 0.4 and for nodules at agreement level 3. Displays error bars with 1 standard deviation

5.9 Performance Comparison with Other Work

We compare our results to those obtained in [Ric+11; Gol+09; MHR10; Tan+11] where the LIDC-IDRI dataset has also been used for validation. Table 5.2 describes the number of true positives in relation to the number of nodules in the validation set, sensitivity, and FPs per scan values of our CADe system ($e3\_r80$) compared to four other studies. The CADe systems described in [Gol+09; Tan+11] were validated on an independent validation set, while the CADe systems described in [Ric+11] and [MHR10] were validated using a 2-fold and a 7-fold cross-validation, respectively. As mentioned above, our CADe system is validated using 5-fold cross-validation.

Our results show a 2.1% improvement in sensitivity compared to the leading result in other studies, while leaving the average number of false positive per scan fixed to 4. Even when considering the 3% standard deviation estimate, which is shown in Figure 5.9 and is attributed to the selection of the complementary training and validation sets, DeepCADe

Table 5.2: Summary of results for five CADe systems.

Study	True positives	Sensitivity	FP per scan	Agreement level
DeepCADe	250/279	89.6%	4	3
Tan et al. [Tan+11]	70/80	87.5%	4	4
Messay et al. [MHR10]	118/143	82.7%	3	1
Golosio et al. [Gol+09]	30/38	79%	4	4
Riccardi et al. [Ric+11]	83/117	71%	6.5	4

demonstrates higher sensitivity compared to the last three studies in Table 5.2, and is showing comparable performance to Tan et al. [Tan+11]. It would have been more fair to compare the results of these studies if they all used the same training and validation sets, or if at least they all reported an estimate to how sensitive their systems are to the selection of the training and validation sets.

In addition to the increase in sensitivity, DeepCADe is validated on a significantly larger number of lung nodules compared to other studies. More specifically, it is validated on 279 lung nodules, while other studies are validated on between 38 and 143 lung nodules. This increases the variation in the appearance of the various nodules in the validation set, and therefore makes their detection by a CADe system more challenging.

The results of the 5-fold cross validation assume a 80/20 division of the 1018 images in the LIDC-IDRI dataset in order to create the training and validation sets. In contrast, all other experiments described in this chapter assume a 90/10 split of the dataset. As in the case of exploring one meta-parameter value at a time, this is done due to the high computational demands, which are required to train the weights and evaluate the performance of a single network.

5.10 Visualizing the annotated nodules and nodule predictions

Here, we visualize a number of CT slice images which illustrates the overlap between lung nodules at agreement level 3 (i.e. nodules that were annotated by at least 3 radiologists) and nodule predictions made by $e3\_r80$. These images are extracted from the validation set, which means they have not been processed by the network during its training.

Figures 5.10 and 5.11 illustrate a number of true positive predictions made by $e3\_r80$. The nodule predictions are marked by a red polygon which is composed of right angles only, and the annotated nodules are marked by a green boundary. It shows that while some nodule predictions are larger than their corresponding annotations, others are very close to the boundary of the annotated nodules. As mentioned in Section 5.5, achieving compact predictions is essential for incorporating CADe systems in the workflow of radiologists since it would prevent them from having to manually scan the predicted areas made by the CADe system, and therefore help gain their trust in the system.

The nodule annotations shown in Figures 5.10 and 5.11 have a wide range of sizes and locations within the lungs, and while some are quite large and visually clear 5.10, others are very small and hard to detect 5.11 which makes them prone to be overlooked by radiologists. In contrast, Figure 5.12 illustrates two lung nodules, which were annotated by at least three radiologists but were not detected by $e3\_r80$. These nodules are indeed very small and hard to detect.

Finally, Figure 5.13 illustrates a number of false positive predictions made by $e3\_r80$. These are nodule predictions that are made by DeepCADe but are false. Having a low number of false positive predictions per scan is also crucial since otherwise the user of the CADe system will have to manually examine many false predictions and therefore lose its trust in the system.

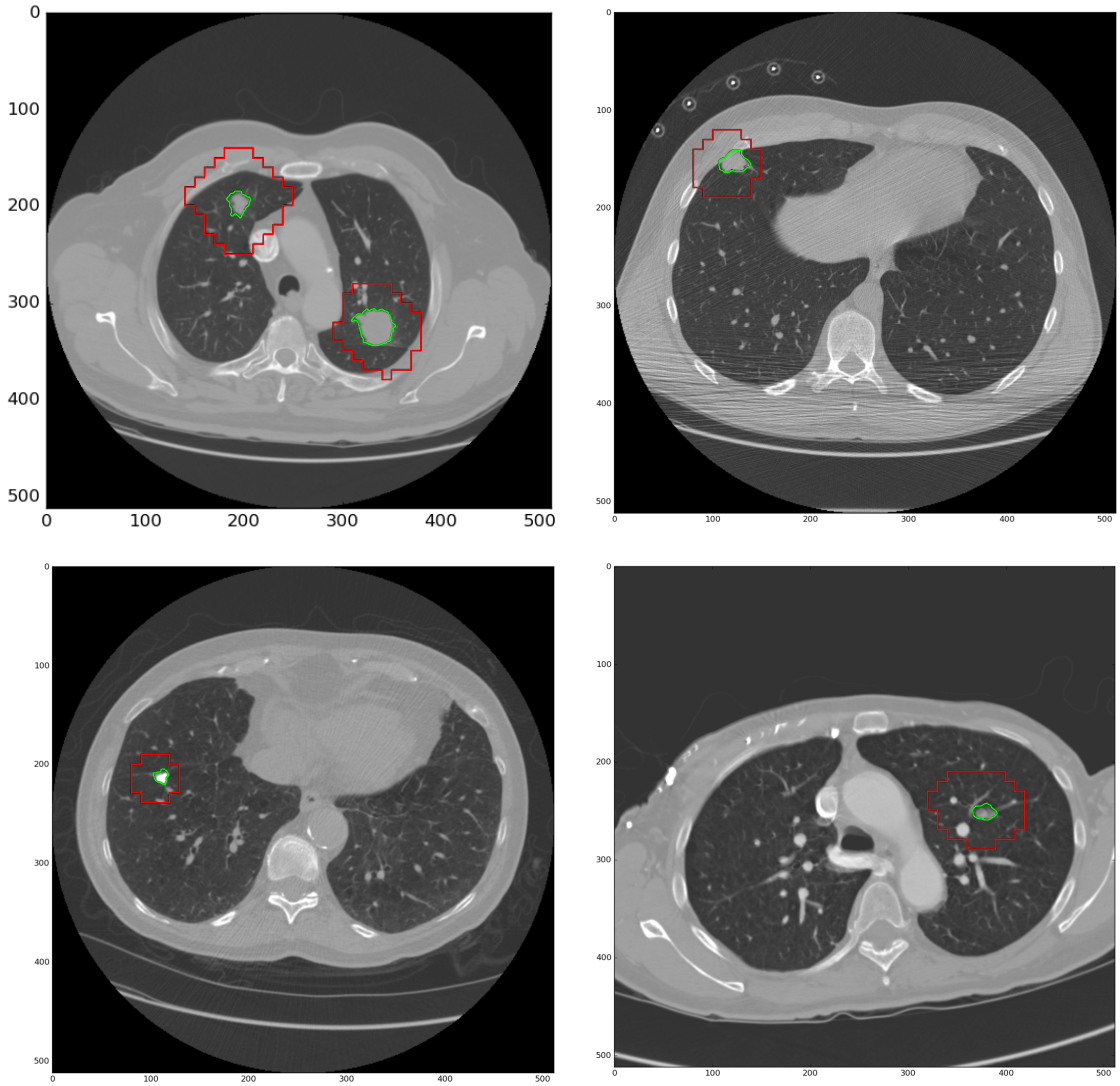

Figure 5.10: A number of true positive predictions made by $e3\_r80$. The nodule predictions are marked by a red polygon which is composed of right angles only, and the annotated nodules are marked by a green boundary.

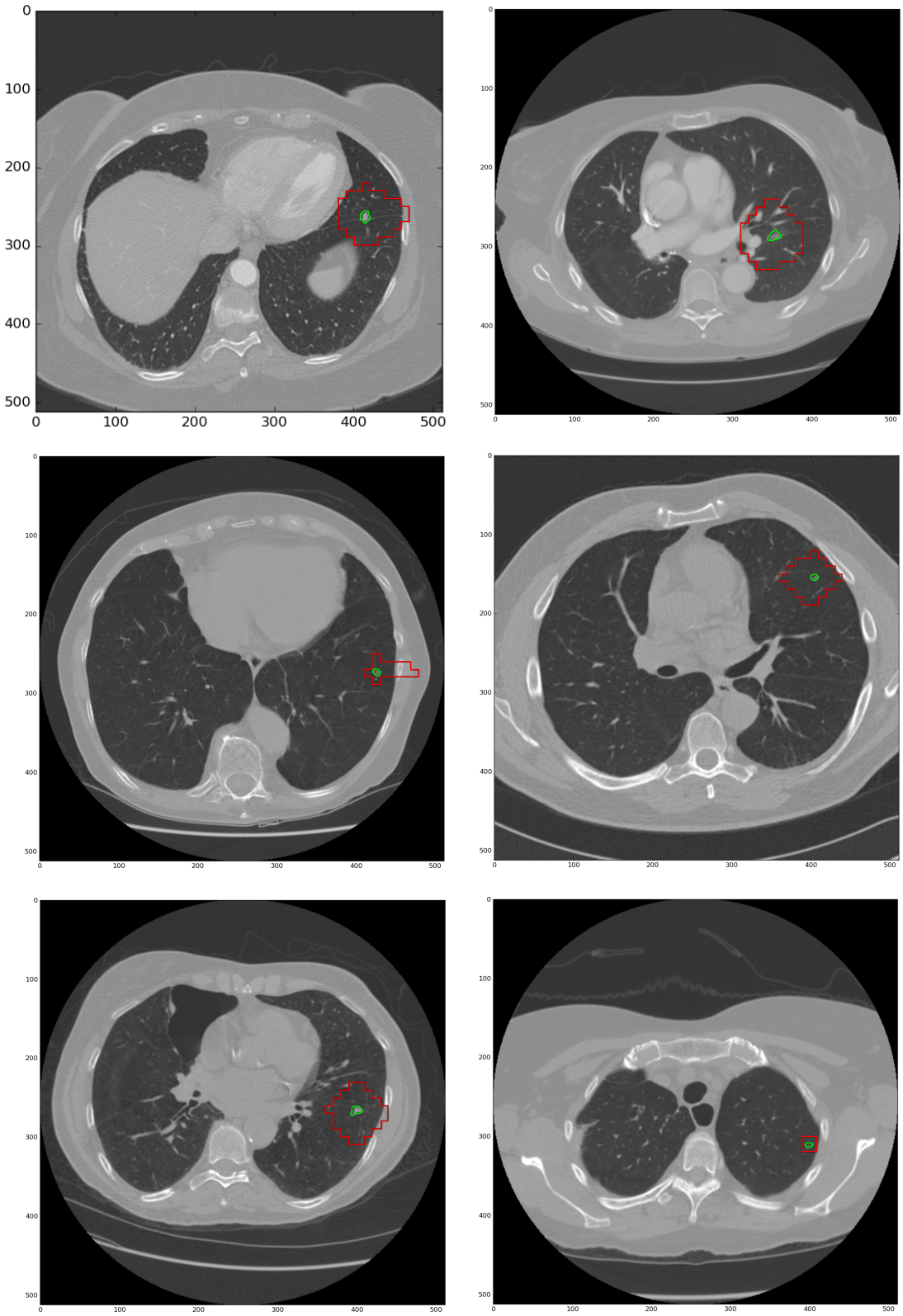

Figure 5.11: A number of true positive predictions made by $e3\_r80$. The nodule predictions are marked by a red polygon which is composed of right angles only, and the annotated nodules are marked by a green boundary.

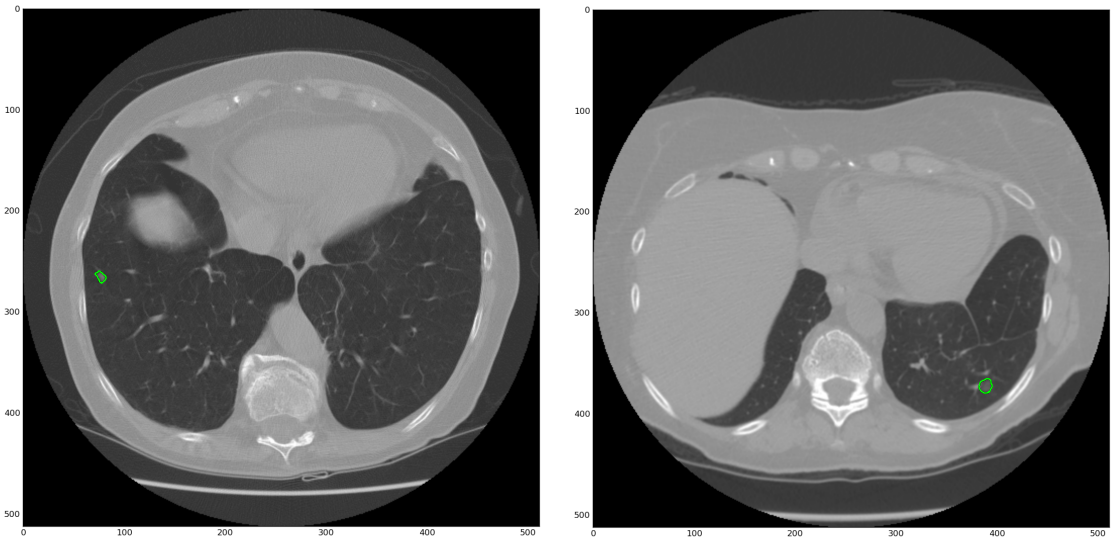

Figure 5.12: Two lung nodules which were annotated by at least three radiologists but were not detected by $e3\_r80$. The annotated nodules are marked by a green boundary.

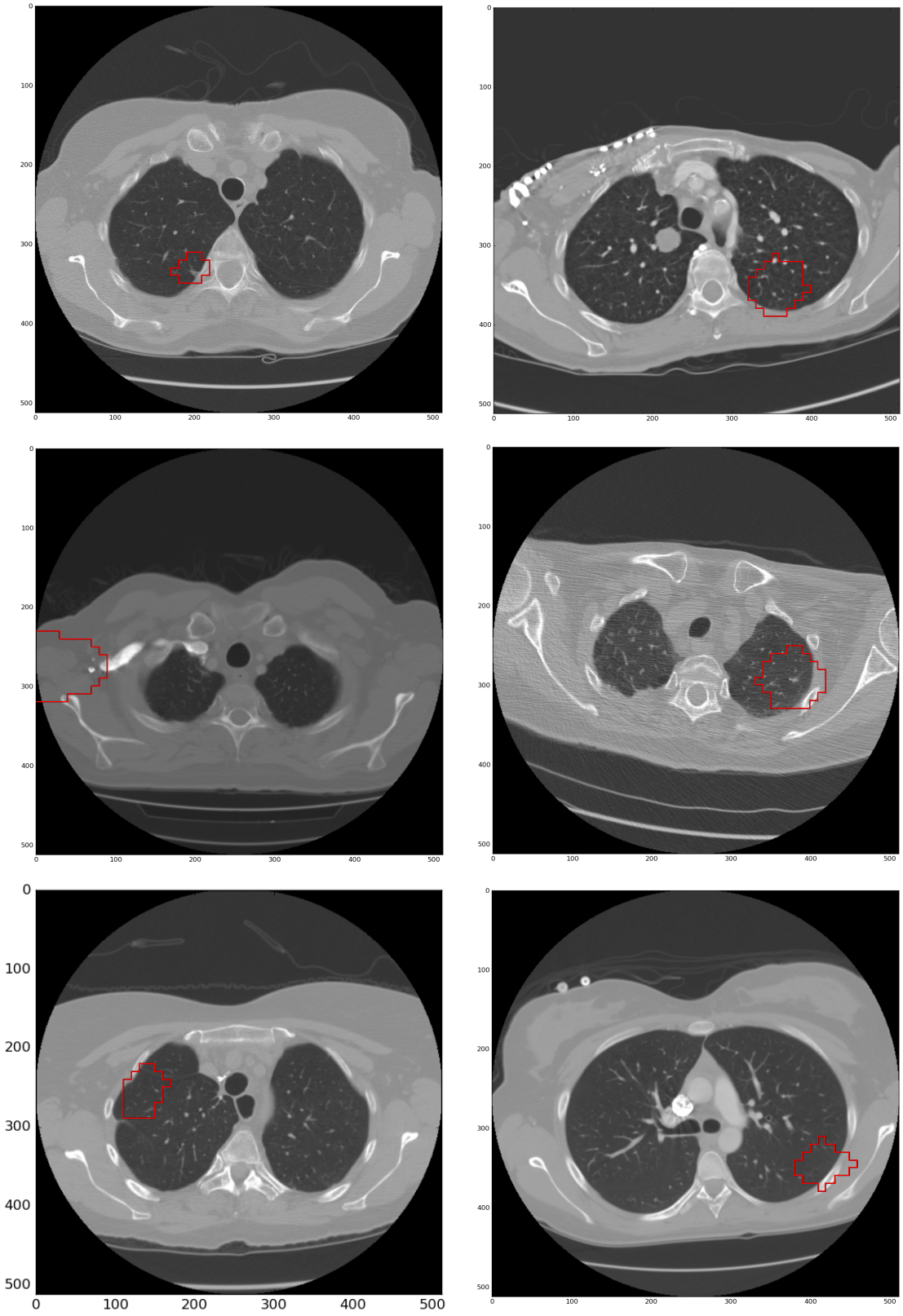

Figure 5.13: A number of false positive predictions made by $e3\_r80$. The nodule predictions are marked by a red polygon which is composed of right angles only.

Chapter 6

Conclusion and Future Work

Early detection of lung nodules in thoracic Computed Tomography (CT) scans is of great importance for the successful diagnosis and treatment of lung cancer. Due to improvements in screening technologies, and an increased demand for their use, radiologists are required to analyze an ever increasing amount of image data, which can affect the quality of their diagnoses. Computer-Aided Detection (CADe) systems are designed to assist radiologists in this endeavor.

In this PhD thesis, we presented DeepCADe, a novel CADe system for the detection of lung nodules in thoracic CT scans which produces improved results compared to the state-of-the-art in this field of research. DeepCADe is based on (1) the publicly available Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) database, which contains 1018 thoracic CT scans with nodules of different shape and size, and (2) a Deep Convolutional Neural Network (DCNN), which is trained using the backpropagation algorithm to extract valuable volumetric features from the input data and detect lung nodules in sub-volumes of CT images.

Considering only lung nodules that have been annotated by at least three radiologists, DeepCADe achieves a 2.1% improvement in sensitivity (true positive rate) over the best result in the current published scientific literature, assuming an equal number of false positives (FPs) per scan. More specifically, it achieves a sensitivity (true positive rate) of 89.6% with 4 FPs per scan, or a sensitivity of 92.8% with 10 FPs per scan. Furthermore, DeepCADe was validated on a larger number of lung nodules compared to other studies. This increases the variation in the appearance of lung nodules and therefore makes their detection by a CADe system more challenging. More specifically, DeepCADe was validated on 279 annotated lung

nodules, while other studies were validated on between 38 and 143 annotated lung nodules.

The practical meaning of this depends on the quality of data the system has been validated on, which can be measured in terms of the variability it encompasses. For example, if the validation set contains images that were acquired by various manufacturers and covers a range of pathologies, a 2.1% increase in performance might be significant and mean that the system has gained the capacity to handle a new set of input examples; however, if no such variability exists in the validation set, an increase of 2.1% might not be as significant. Additional data is required to assess the variability in the validation set.

We explored some of the meta parameters that affect the performance of DeepCADe and identified which ones have the most impact. The meta parameters we explored include (1) the DCNN architecture, i.e. its structure in terms of convolution layers, fully-connected layers, pooling layers, and activation functions, (2) the receptive field of the network, which defines the dimensions of its input, i.e. how much of the CT scan is processed by the network in a single forward pass, (3) a threshold value, which affects the sliding window algorithm with which the network is used to detect nodules in complete CT scans, and (4) the agreement level, which is used to interpret the independent nodule annotations of four experienced radiologists.

The receptive field of the network turned out to have a significant effect on the performance of DeepCADe. As shown in Figure 5.5, we learned that using a receptive field of size $60 \times 60 \times 5$ ($e3\_r60$) leads to a significant improvement in sensitivity compared to using a receptive field of $20 \times 20 \times 5$ ($e3\_r20$). More specifically, assuming the system is configured to allow 5 false positives (FPs) per scan, $e3\_r60$ produces an absolute 25% improvement in sensitivity compared to $e3\_r20$. We hypothesize this positive effect on results to be due to the network's capacity to process a larger region of the CT scan at a time, and therefore, have a larger context to work with.

In summary, the key contributions of this work are:

1. DeepCADe, a novel CADe system for the detection of lung nodules in CT scans, which is based on a deep learning architecture and a sliding window algorithm.

2. state-of-the-art results on the largest publicly available dataset of annotated lung nodules in thoracic CT scans, namely the LIDC-IDRI dataset.

3. demonstrating that DCNNs, which have shown tremendous results in other computer vision tasks, can also be successfully applied to the detection of lung nodules in CT scans.

4. evidence that volumetric features, which are trained through the backpropagation algorithm, are more useful to a classifier compared to hand-engineered ones.

5. an in-depth discussion on the meta-parameter space of DeepCADe, and of DCNNs in general.

6. a single comprehensive text on the detection of lung nodules in thoracic CT scans, which is accessible to both computer scientists and radiologists.

In order for DeepCADe and other CADe systems to become fully integrated into the daily practice of radiologists further research is required. Here, we identify a number of challenges and research avenues which are of great importance for making this transformation a reality.

First, while the LIDC-IDRI dataset has been very instrumental for us and other research groups in designing and validating our CADe systems, it is still limited in terms of the number of images it contains and the degree by which these images vary from one another. Having positive results on the LIDC-IDRI dataset, even if such results have been produced using cross-validation, still does not guarantee the system will perform the same in real world

settings. The variability of CT images that radiologists encounter in their daily practice is significant, and any CADe system, which aims at becoming fully integrated into radiology practice, must take this into consideration. The sources for this variability range from the use of different CT scanners to various clinical states of the lungs.

Having access to annotated datasets that are larger and that account for this variability would indeed help in overcoming this issue. However, this would require the medical field to be more open to share its resources while still performing its important duty of protecting the privacy of patients. Alternatively, augmenting the data that is already publicly available is also expected to be of great importance. Data augmentation has already shown to be beneficial in other computer vision tasks [KSH12; CMS12; NHH15; Che+14a; LSD15], and finding creative ways to augment annotated CT scans is expected to have a similar effect towards making CADe systems more robust to variations in CT scans.

Second, while DeepCADe can detect the location of multiple lung nodules without predicting excessive amounts of false positives, and while it does provide the user with a rough estimate to where the boundaries of these nodule predictions are, DeepCADe does not output the exact boundaries of nodule predictions. Recent advances in the field of semantic segmentation [NHH15; Che+14a; LSD15] can prove to be effective here. These include the use of Fully Convolutional Networks (FCNs) to do the mapping between images and their corresponding masks. However, FCNs are very demanding in terms of computation so applying them to complete CT scans is not feasible with modern computer hardware. DeepCADe can assist here by accurately detecting the location of lung nodules, which can then be used by a FCN to perform the segmentation. This way the input for the FCN is of a manageable size, and the entire computation from image to exact segmentation of all detected nodules can be done in a reasonable amount of time.

Finally, the meta-parameters of CADe systems have shown to have a significant impact on the performance of such systems, and therefore finding the optimal set of meta-parameter

values is of utmost importance. One approach of achieving that, which we embraced in this work, is to do so manually. While this approach can be effective, it is very time-consuming and has the danger of having a designer bias towards known architectures. Alternatively, finding the optimal set of meta-parameters can be automated. Further research should be focused on this. Evolutionary algorithms are one approach that comes to mind when thinking about this problem [Rea+17], but one must remember that evaluating a single set of meta-parameters can be very computationally expensive, and therefore might not be an ideal choice for a fitness function which is executed many times as part of the evolutionary process.

While Deep Convolutional Neural Networks have shown tremendous results in other computer vision tasks, our work demonstrates that these powerful machine learning models can also be successfully applied to the detection of lung nodules in thoracic CT scans. Given a sufficient amount of high quality data, both in terms of quantity and variability, CADe systems such as DeepCADe can help transform radiology into a more accurate and accessible medical practice, and make use of the enormous amounts of annotated medical imaging data that is being generated everyday by radiologists.

Appendix A

Network Structures

The architecture of a DCNN can be described by listing its layers in sequence from input to output. For example, here is a DCNN architecture we explored in this thesis (denoted as $e3$): $5\times20\times20-128C3\times9\times9-MP1\times2\times2-256C2\times4\times4-MP2\times2\times2-512C1\times3\times3-512C1\times3\times3-MP1\times2\times2-2709FC-2709FC-2709FC-1FC$, where, for example, 5×20×20 is the input layer (the receptive field of the DCNN), $128C3\times9\times9$ represents a volumetric convolution layer with 128 feature kernels of size 3×9×9, $MP1\times2\times2$ represents a max pooling layer with a kernel of size 1×2×2, and $2709FC$ represents a fully-connected layer with 2709 units.

Other meta-parameters that are not included in this description are (1) the stride value of both the volumetric convolution and max-pooling layers, which is set to either 1 or 2 in all the experiments we conducted, (2) the stride value used for computing the voting grid, which is set to 1 for the depth axis of a CT scan and 10 for the in-plane axes of a scan, (3) the activation functions that are applied after every convolutional and fully connected layer, and (4) the zero padding size. These meta-parameters are determined so to ensure that the number of units at the beginning of the classifier module of the CNN is exactly 2,055. Ensuring an equal number of inputs to the classifier module of the CNN allows for a more fair comparison between the various experiments we performed. Also, in addition to the activations of its previous layer, the first fully-connected layer of every CNN receives 7 additional values which represent positional information of the receptive field in relation to the entire CT image and information regarding the DICOM image.

Here are all the network architectures examined in this thesis:

$e1$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 9850FC - 1FC$

$e2$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 3588 - 3588 - 1FC$

$e3 = e3\_2709 = e3\_4Conv = e3\_relu = e3\_r20$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e4$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2274FC - 2274FC - 2274FC - 2274FC - 1FC$

$e3\_500$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 500FC - 500FC - 500FC - 1FC$

$e3\_1000$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 1000FC - 1000FC - 1000FC - 1FC$

$e3\_1500$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 1500FC - 1500FC - 1500FC - 1FC$

$e3\_0Conv$:

$5 \times 20 \times 20 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_2Conv$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 512C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_6Conv$:

$5 \times 20 \times 20 - 128C3 \times 9 \times 9 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_sig$:

Same as $e3$ except that all activation functions are set to a sigmoid function instead of a ReLU function.

$e3\_tanh$:

Same as $e3$ except that all activation functions are set a hyperbolic tangent function instead of a ReLU function.

$e3\_r40$:

$5 \times 40 \times 40 - 128C3 \times 12 \times 12 - MP1 \times 2 \times 2 - 256C2 \times 7 \times 7 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_r60$:

$5 \times 60 \times 60 - 128C3 \times 10 \times 10 - MP1 \times 2 \times 2 - 256C2 \times 6 \times 6 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_r80$:

$5 \times 80 \times 80 - 128C3 \times 10 \times 10 - MP1 \times 2 \times 2 - 256C2 \times 4 \times 4 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

$e3\_r60.10$:

$10 \times 60 \times 60 - 128C4 \times 10 \times 10 - MP2 \times 2 \times 2 - 256C2 \times 6 \times 6 - MP2 \times 2 \times 2 - 512C1 \times 3 \times 3 - 512C1 \times 3 \times 3 - MP1 \times 2 \times 2 - 2709FC - 2709FC - 2709FC - 1FC$

Bibliography

[Lit+17] Geert Litjens et al. "A survey on deep learning in medical image analysis". In: *arXiv preprint arXiv:1702.05747* (2017).

[Rea+17] Esteban Real et al. "Large-scale evolution of image classifiers". In: *arXiv preprint arXiv:1703.01041* (2017).

[SWS17] Dinggang Shen, Guorong Wu, and Heung-Il Suk. "Deep learning in medical image analysis". In: *Annual Review of Biomedical Engineering* 0 (2017).

[Ani+16] Rushil Anirudh et al. "Lung nodule detection using 3D convolutional neural networks trained on weakly labeled data". In: *SPIE Medical Imaging*. International Society for Optics and Photonics. 2016, pp. 978532–978532.

[CCI16] T Colin Campbell and Thomas M Campbell II. *The China Study: Revised and Expanded Edition: The Most Comprehensive Study of Nutrition Ever Conducted and the Startling Implications for Diet, Weight Loss, and Long-Term Health*. BenBella Books, Inc., 2016.

[GJD16] Rotem Golan, Christian Jacob, and Jörg Denzinger. "Lung nodule detection in CT images using deep convolutional neural networks". In: *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE. 2016, pp. 243–250.

[GG16] Richard Gruetzemacher and Ashish Gupta. "Using Deep Learning for Pulmonary Nodule Detection & Diagnosis". In: (2016).

[Shi+16] Hoo-Chang Shin et al. "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning". In: *IEEE transactions on medical imaging* 35.5 (2016), pp. 1285–1298.

[IS15] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International Conference on Machine Learning*. 2015, pp. 448–456.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[Mai+15] Rafael Simon Maia et al. "An algorithm for noise correction of dual-energy computed tomography material density images". In: *International journal of computer assisted radiology and surgery* 10.1 (2015), pp. 87–100.

[McD+15] Robert J McDonald et al. "The effects of changes in utilization and technological advancements of cross-sectional imaging on radiologist workload". In: *Academic radiology* 22.9 (2015), pp. 1191–1198.

[NHH15] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. "Learning deconvolution network for semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1520–1528.

[SMJ15] Rebecca L Siegel, Kimberly D Miller, and Ahmedin Jemal. "Cancer statistics, 2015". In: *CA: a cancer journal for clinicians* 65.1 (2015), pp. 5–29.

[Tea15] The Cancer Imaging Archive Team. *Data From LIDC-IDRI*. 2015. URL: `http://dx.doi.org/10.7937/K9/TCIA.2015.LO9QL9SX`.

[Xu+15] Bing Xu et al. "Empirical evaluation of rectified activations in convolutional network". In: *arXiv preprint arXiv:1505.00853* (2015).

[Che+14a] Liang-Chieh Chen et al. "Semantic image segmentation with deep convolutional nets and fully connected crfs". In: *arXiv preprint arXiv:1412.7062* (2014).

[Che+14b] Sharan Chetlur et al. "cudnn: Efficient primitives for deep learning". In: *arXiv preprint arXiv:1410.0759* (2014).

[Ng14] Andrew Ng. *Online course (coursera): Machine Learning*. 2014. URL: https://www.coursera.org/learn/machine-learning/.

[Rus+14] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International Journal of Computer Vision* (2014), pp. 1–42.

[Sri+14] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting." In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[Sze+14] Christian Szegedy et al. "Going deeper with convolutions". In: *arXiv preprint arXiv:1409.4842* (2014).

[BCV13] Yoshua Bengio, Aaron Courville, and Pierre Vincent. "Representation learning: A review and new perspectives". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.8 (2013), pp. 1798–1828.

[Cha13] Dev P Chakraborty. "A brief history of free-response receiver operating characteristic paradigm data analysis". In: *Academic radiology* 20.7 (2013), pp. 915–919.

[Cla+13] Kenneth Clark et al. "The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository". In: *Journal of digital imaging* 26.6 (2013), pp. 1045–1057.

[Le13] Quoc V Le. "Building high-level features using large scale unsupervised learning". In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE. 2013, pp. 8595–8598.

[ODS13] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. "Deep content-based music recommendation". In: *Advances in Neural Information Processing Systems*. 2013, pp. 2643–2651.

[Suz13] Kenji Suzuki. "Machine Learning in Computer-Aided Diagnosis of the Thorax and Colon in CT: A Survey". In: *IEICE transactions on information and systems* 96.4 (2013), pp. 772–783.

[CMS12] Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. "Multi-column deep neural networks for image classification". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 3642–3649.

[Den12] Li Deng. "The MNIST database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[Hin+12a] Geoffrey Hinton et al. "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups". In: *Signal Processing Magazine, IEEE* 29.6 (2012), pp. 82–97.

[Hin+12b] Geoffrey E Hinton et al. "Improving neural networks by preventing co-adaptation of feature detectors". In: *arXiv preprint arXiv:1207.0580* (2012).

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[LeC+12] Yann A LeCun et al. "Efficient backprop". In: *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

[AI+11] Samuel G Armato III et al. "The lung image database consortium (LIDC) and image database resource initiative (IDRI): a completed reference database of lung nodules on CT scans". In: *Medical physics* 38.2 (2011), pp. 915–931.

[Kra+11] Barnett S Kramer et al. "Lung cancer screening with low-dose helical CT: results from the National Lung Screening Trial (NLST)". In: *Journal of medical screening* 18.3 (2011), pp. 109–111.

[Ric+11] Alessandro Riccardi et al. “Computer-aided detection of lung nodules via 3D fast radial transform, scale space representation, and Zernike MIP classification”. In: *Medical physics* 38.4 (2011), pp. 1962–1971.

[Sta+11] Johannes Stallkamp et al. “The German traffic sign recognition benchmark: a multi-class classification competition”. In: *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE. 2011, pp. 1453–1460.

[Tan+11] Maxine Tan et al. “A novel computer-aided lung nodule detection system for CT images”. In: *Medical physics* 38.10 (2011), pp. 5630–5645.

[Liu+10] Cheng-Lin Liu et al. “Chinese handwriting recognition contest 2010”. In: *Pattern Recognition (CCPR), 2010 Chinese Conference on*. IEEE. 2010, pp. 1–5.

[MHR10] Temesguen Messay, Russell C Hardie, and Steven K Rogers. “A new computationally efficient CAD system for pulmonary nodule detection in CT imagery”. In: *Medical Image Analysis* 14.3 (2010), pp. 390–406.

[SSB10] Igor Shuryak, Rainer K Sachs, and David J Brenner. “Cancer risks after radiation exposure in middle age”. In: *Journal of the National Cancer Institute* 102.21 (2010), pp. 1628–1636.

[SB+10] Rebecca Smith-Bindman et al. “Is computed tomography safe”. In: *N Engl j Med* 363.1 (2010), pp. 1–4.

[Ben09] Yoshua Bengio. “Learning deep architectures for AI”. In: *Foundations and trends® in Machine Learning* 2.1 (2009), pp. 1–127.

[Den+09] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 248–255.

[Faz+09] Reza Fazel et al. “Exposure to low-dose ionizing radiation from medical imaging procedures”. In: *New England Journal of Medicine* 361.9 (2009), pp. 849–857.

[Gol+09] Bruno Golosio et al. “A novel multithreshold method for nodule detection in lung CT”. In: *Medical physics* 36.8 (2009), pp. 3607–3618.

[Gon+09] Amy Berrington de González et al. “Projected cancer risks from computed tomographic scans performed in the United States in 2007”. In: *Archives of internal medicine* 169.22 (2009), pp. 2071–2077.

[KH09] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. 2009.

[Lee+09] Honglak Lee et al. “Unsupervised feature learning for audio classification using convolutional deep belief networks.” In: *NIPS*. Vol. 9. 2009, pp. 1096–1104.

[SB+09] Rebecca Smith-Bindman et al. “Radiation dose associated with common computed tomography examinations and the associated lifetime attributable risk of cancer”. In: *Archives of internal medicine* 169.22 (2009), pp. 2078–2086.

[Tan+09] Maxine Tan et al. “Automated feature selection in neuroevolution”. In: *Evolutionary Intelligence* 1.4 (2009), pp. 271–292.

[CW08] Ronan Collobert and Jason Weston. “A unified architecture for natural language processing: Deep neural networks with multitask learning”. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 160–167.

[DPC08] James J DiCarlo, Nicolas Pinto, and David Daniel Cox. “Why is Real-World Visual Object Recognition Hard?” In: (2008).

[Tie08] Tijmen Tieleman. “Training restricted Boltzmann machines using approximations to the likelihood gradient”. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 1064–1071.

[TM+08] Tinne Tuytelaars, Krystian Mikolajczyk, et al. "Local invariant feature detectors: a survey". In: *Foundations and trends® in computer graphics and vision* 3.3 (2008), pp. 177–280.

[Ser+07] Thomas Serre et al. "A quantitative theory of immediate visual recognition". In: *Progress in brain research* 165 (2007), pp. 33–56.

[HS06] Geoffrey E Hinton and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *Science* 313.5786 (2006), pp. 504–507.

[Bus05] Isabel Bush. "Lung Nodule Detection and Classification". In: (2005).

[LHB04] Yann LeCun, Fu Jie Huang, and Leon Bottou. "Learning methods for generic object recognition with invariance to pose and lighting". In: *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2004, pp. II–97.

[LZ03] Gareth Loy and Alexander Zelinsky. "Fast radial symmetry for detecting points of interest". In: *IEEE Transactions on pattern analysis and machine intelligence* 25.8 (2003), pp. 959–973.

[CBM02] Ronan Collobert, Samy Bengio, and Johnny Mariéthoz. *Torch: a modular machine learning software library*. Tech. rep. IDIAP, 2002.

[Gru+02] James F Gruden et al. "Incremental benefit of maximum-intensity-projection images on observer detection of small pulmonary nodules revealed by multidetector CT". In: *American Journal of Roentgenology* 179.1 (2002), pp. 149–157.

[SM02] Kenneth O Stanley and Risto Miikkulainen. "Evolving neural networks through augmenting topologies". In: *Evolutionary computation* 10.2 (2002), pp. 99–127.

[FSA99] Yoav Freund, Robert Schapire, and N Abe. "A short introduction to boosting". In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999), p. 1612.

[LeC+98] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[Bre96] Leo Breiman. "Bagging predictors". In: *Machine learning* 24.2 (1996), pp. 123–140.

[Gro95] Patrick J Grother. "NIST special database 19 handprinted forms and characters database". In: *National Institute of Standards and Technology* (1995).

[Koh+95] Ron Kohavi et al. "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Ijcai*. Vol. 14. 2. Stanford, CA. 1995, pp. 1137–1145.

[AB94] Rolf Adams and Leanne Bischof. "Seeded region growing". In: *IEEE Transactions on pattern analysis and machine intelligence* 16.6 (1994), pp. 641–647.

[LeC+89] Yann LeCun et al. "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4 (1989), pp. 541–551.

[KH88] ALIREZA Khotanzad and Yaw Hua Hong. "Rotation invariant pattern recognition using Zernike moments". In: *Pattern Recognition, 1988., 9th International Conference on*. IEEE. 1988, pp. 326–328.

[RHW88] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *Cognitive modeling* 5 (1988), p. 3.

[Fuk80] Kunihiko Fukushima. "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". In: *Biological cybernetics* 36.4 (1980), pp. 193–202.

[Bun+77] Philip C Bunch et al. “A free response approach to the measurement and characterization of radiographic observer performance”. In: *Application of Optical Instrumentation in Medicine VI*. International Society for Optics and Photonics. 1977, pp. 124–135.

[Mil69] Harold Miller. “The FROC curve: a representation of the observer’s performance for the method of free response”. In: *The Journal of the Acoustical Society of America* 46.6B (1969), pp. 1473–1476.

[HW62] David H Hubel and Torsten N Wiesel. “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”. In: *The Journal of physiology* 160.1 (1962), p. 106.

[Tur50] Alan M Turing. “Computing machinery and intelligence”. In: *Mind* 59.236 (1950), pp. 433–460.

[Fis36] Ronald A Fisher. “The use of multiple measurements in taxonomic problems”. In: *Annals of eugenics* 7.2 (1936), pp. 179–188.

[Fac] *Facebook*. URL: `https://www.facebook.com`.

[You] *YouTube*. URL: `https://www.youtube.com`.