Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

Lung Opacity Detection With Convolutional Neural Networks Using Chest X-rays

Author

Khan Fashee Monowar

Roll No. 1503084

Department of Computer Science & Engineering

Rajshahi University of Engineering & Technology

Supervised by

Nahin Ul Sadad

Lecturer

Department of Computer Science & Engineering

Rajshahi University of Engineering & Technology

ACKNOWLEDGEMENT

First of all, I would like to praise and show gratitude to the almighty Allah (SWT) for giving me the strength to complete this task. Without his grace, it would not be possible for me.

I would like to express my sincerest gratitude and respect to my supervisor **Nahin Ul Sadad** , Lecturer, Department of Computer Science & Engineering, Rajshahi University of Engineering & Technology, for his valuable advice and directions on this research. The step by step progression of the research work has been possible only for his great supervision. I was really fortunate to have his constant support, inspiration and patience throughout the whole procedure.

I am also very grateful to **Dr. Al Mehedi Hasan**, Professor, Department of Computer Science Engineering, Rajshahi University of Engineering Technology, and all other respected faculty members of Department of Computer Science & Engineering, Rajshahi University of Engineering & Technology, for their inspiration, endless facilitation and suggestions.

Finally, I would like to express my gratitude to my beloved parents and family members for their blessings and support as well as my peers and all other well-wishers for their kind co-operation.

Date: March, 2021 Khan Fashee Monowar

RUET, Rajshahi

Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

CERTIFICATE

*This is to certify that this thesis report entitled "**Lung Opacity Detection With Convolutional Neural Networks Using Chest X-rays**" submitted by **Khan Fashee Monowar**, Roll. 1503084, in partial fulfillment of the requirement for the award of the degree of Bachelor of Science in Computer Science & Engineering of Rajshahi University of Engineering & Technology, Bangladesh is a record of the candidate's own work carried out by him under my supervision. This thesis has not been submitted for the award of any other degree.*

Supervisor	**External Examiner**
____________________	____________________
Nahin Ul Sadad	**Biprodip Pal**
Lecturer	Assistant Professor
Department of Computer Science & Engineering	Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology	Rajshahi University of Engineering & Technology
Rajshahi-6204	Rajshahi-6204

ABSTRACT

Chest X-ray interpretation is very crucial to detect cardiothoracic and pulmonary abnormalities. This time-consuming and tedious task should be error-free, fast, and reliable otherwise a single mistake may cause serious harm to patients. Recently, deep learning achieves radiologist-level performance in chest X-ray interpretation. A CAD (Computer-aided detection system) can help radiologists to review chest X-rays fast and accurately. In our research, we trained and evaluated various deep convolutional neural networks (CNN) architectures to detect potential lung opacity from chest X-rays. We observed how strongly architectures could differentiate lung opacity from normal and other abnormal chest X-rays. In these circumstances, A CNN based model (Xception) achieved 91.0% AUC along with 83.95% accuracy. Moreover, we also observed models achieved a better performance on lung opacity vs normal chest X-ray classification (excluding abnormal class) where Xception achieved 99.1% AUC, 97.19% sensitivity, and 95.71%accuracy. Therefore, the purpose of this study is to investigate the classification ability of deep CNN architectures which helps to develop an automatic lung opacity detection system.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Overview

This chapter begins with the problem statement. Then motivations behind this thesis topic, objectives are described. Then in the literature review section, some recent research works have been described shortly. Then, in thesis contribution, contributions of the thesis are outlined. Finally, the chapter ends with a conclusion .

1.2 Problem Statement

Lung diseases and pulmonary abnormalities are one of the leading causes of mortality all over the world. According to the National Center For Health Statistics, nearly 150 thousand people died of lung cancer in the USA in 2016 [11] and more than 33 million suffer from chronic lung diseases. Pneumonia is another pulmonary disease caused by bacterial or viral infection in the lungs. It causes pulmonary opacities in the lungs and has higher death rates than lung cancer. In 2017, approximately 2.56 million people died of pneumonia including 800,000 children aged less than five years. Moreover, Scientists and physicians can't reduce the overall mortality rate significantly since 1990[32]. The death rate is even higher in underdeveloped countries. In Bangladesh, More than 38,500 people lost their lives due to pneumonia in 2017. Sub-Saharan and Southeast Asia have the highest mortality rate in the world[11].

Generally, expert radiologists diagnosis chest x-ray to identify lung opacities effectively. The process is quite time-consuming and leads to fatigue-based diagnostic errors. While reviewing, a single mistake can create a significant negative impact on a patient's health. Goddart et al.[14] reported that the prevalence of clinically significant errors in radiology was in the range of 2 to 20 percent in 2001. Lack of adequate knowledge, inaccurate imaging techniques, workloads are some of the reasons for these errors.

1.3 Motivation

Lung opacity related diseases are the largest infectious cause of death in human worldwide. In Bangladesh, almost 20,000 children aged less than 5 years, died of pneumonia in 2017 . Scientists further warn 140,000 children may die of pneumonia in the next decade if proper steps are not taken care of[1]. Shortage of medical resources, low doctor to patient ratio, poverty lead on to a higher death rate in underdeveloped countries. Physicians of these countries need to diagnose scads of Chest X-rays daily which prone to incorrect diagnostic results. According to the care quality commission, Radiologist or clinician failed to formally review a total of 23,000 chest X-rays in 2016-2017 at Queen Alexandra Hospital in the United Kingdom[5]. Moreover, three patients with lung cancer suffered significant harm because their chest X-rays had not been properly assessed. In these circumstances, deep learning based Computer-aided detection (CAD) technology can help physicians diagnosing Chest X-rays image accurately. CAD decreases observational oversight and the false-negative rate[5]. These motivates us to conduct our research to detecting lung opacity related diseases from chest x-rays which will be help doctors to accurately diagnosis diseases. We are also motivated to help future researchers by doing a comparison between top CNN architectures to detect lung opacity.

1.4 Research Objectives

With the clarification of problem statement, this literature can now move on to pointing out the research objectives of this work. The objectives are followed below:

- Preprocess chest x-ray image with runtime augmentation
- Split the training procedure into 2 phases to do multiclass and binary classification to asses the performacne of top CNN architectures
- Apply different deep convolutional neural networks to detect the presence of lung opacity related diseases in chest x-ray images as accurately as possible.

1.5 Literature Review

In recent years, Deep learning has seen tremendous success in different fields including bioinformatics and medical image analysis. Consequently, researchers get interested to explore the medical domain with deep learning. Irvin et al.[23] already showed their model outperformed radiologist's performance to diagnose multiple common thorax diseases from front-view chest X-rays. Rajpurkar et al.[33] compared their model discriminative performance with the performance of 6 board-certified radiologists based on AUC. Their proposed model, ChexNext achieved comparable performance to practicing radiologists. In the study of [44], Andrew et al. showed a comparative performance of different CNN models to detect moderate and large pneumothorax on frontal chest X-rays achieving a maximum of 98% AUC employing the Inception CNN model with transfer learning. So, deep learning already achieves its superhuman accuracy to detect multiple diseases.

In computer vision, deep learning has already confirmed its superhuman accuracy to image classification[5]. In [18], He et al. showed 3.57% top-5 error rate on ImageNet test set. In the medical image domain, Rajpurkar et al.[34] proposed an algorithm (DenseNet121, transfer learning with fine-tuning) that can detect thorax diseases from chest X-rays at a level exceeding practicing radiologists. Anitha et al.[4] got over 90% test accuracy on brain tumor image classification using a two-tier classifier with an adaptive segmentation technique. Therefore, deep learning models with proper optimization, get a lot of attention to solve the various problem in the medical image domain including classification. In pediatric chest X-ray classification, several research groups have done some decent research works. Stephen et al.[41] proposed a CNN model to automatically perform pediatric chest x-ray image classification. They achieved 95.31% and 93.73% training and validation accuracy respectively. In [28], Liang et al. presented 51 layers CNN architecture by implementing transfer learning methods. Moreover, they compared their proposed model with some off-the-shelf CNN models. Off-the-shelf CNN models refers the pretrained models which are used as feature extractor and the weights of last layer are modified[5]. They achieved 90.5% training accuracy and 95.3% AUC score. Kermany et al. [25] implemented transfer learning on their CNN architecture and achieved 92% training accuracy. In[16], Gu et al. achieved 82.34% AUC score by implementing DCNN with transfer learning.

In lung opacity classification, Yu-Xing er al. classify the RSNA Pneumonia challenge dataset to detect lung opacity. They selected binary class (Lung opacity and Normal) and achieved 98% and 92.2% AUC and sensitivity score respectively. In the image identification localization domain, Jaiswal et al.[24] employed the Mask-RCNN model to identify possible pneumonia in chest X-rays. Gabruseva et al.[13] employed single-shot detectors algorithms to find out pneumonia regions in chest X-rays based on mAP scores. RetinaNet with Se-

ResNext101 encoders pretrained on ImageNet was employed and achieved at least 0.26 validation mAP. Non-Image features or image metadata contain crucial information to detect any diseases. Baltruschat et al.[5] achieved up to 80.6% AUC on the NIH chest X-ray dataset employing the ResNet152 CNN model concatenating pre-processed non-image feature vectors with the flattened layer of the model. In our study, We didn't consider any image metadata for the training procedure.

1.6 Thesis Contribution

The main contributions of the thesis are as follows:

- We found top 5 CNN architectures based on AUC score to classify lung opacity.
- We showed the performance of these architectures on binary and multiclass classification So, future researchers can get help from this comparison study to choose the right architecture to classify lung opacity during building a CAD (Computer-Aided Detection System)
- The models leave opportunities for further improvement and it can be modified for even better results in the future.

1.7 Publication

We wrote a manuscript to publish our research. The manuscript was accepted and successfully presented at the 11th International Conference on Electrical and Computer Engineering (ICECE 2020) which was organized by the Bangladesh University of Engineering Technology (BUET), Dhaka, Bangladesh on 17-19 December 2020.

1.8 Thesis Organization

Chapter 2 - Lung Opacity

This chapter describes the signs and symptoms, fatality, diagnosis and tests of lung opacity related diseases.

Chapter 3 - Neural Networks (NN)

This chapter discusses the architecture, data preprocessing, weights initialization, loss function, regularization and how learning is achieved in neural networks.

Chapter 4 - Convolutional Neural Networks

This chapter discusses the architecture, layers and some known popular convolutional neural network models available right now. It also distinguishes the difference between an ordinary neural network and a convolutional neural network.

Chapter 5 - Data Augmentation

This chapter discusses the necessity and the different methods of data augmentation in this topic.

Chapter 6 - Results and Performance Analysis

This chapter discusses the dataset we worked with and the findings, results, and analysis of the thesis work. It also compares with other works relating to this area.

Chapter 7 - Conclusion and Future Works

This chapter concludes the thesis, describes its limitation and shows a direction of future work.

1.9 Conclusion

Lung opacity related diseases are such diseases that need to be detected early in the stage and accurately so it can be cured. The thesis introduces an automated system which can solve this problem stated.

Chapter 2

Lung Opacity

2.1 Introduction

Pulmonary opacification/lung opacity represents the result of a decrease in the ratio of gas to soft tissue (blood, lung parenchyma and stroma) in the lung. When reviewing an area of increased attenuation (opacification) on a chest radiograph or CT it is vital to determine where the opacification is. The patterns can broadly be divided into airspace opacification, lines and dots. In other words, Opacity refers to any area that preferentially attenuates the x-ray beam and therefore appears more opaque than the surrounding area. It is a nonspecific term that does not indicate the size or pathologic nature of the abnormality[15]. Pneumonia is a lung infection that can be caused by bacteria, viruses, or fungi. Because of the infection and the body's immune response, the sacks in the lungs (termed alveoli) are filled with fluids instead of air. Including, Lung opacities refers to pneumonia-like (pneumonia, infiltration, consolidation) diseases[37]. The reason that pneumonia associated lung opacities look diffuse on the chest radiograph is that the infection and fluid that accumulate spread within the normal tree of airways in the lung. There is no clear border where the infection stops. That is different from other diseases like tumors, which are totally different from the normal lung and do not maintain the normal structure of the airways inside the lung.

2.2 Pneumonia/Pneumonia-related Diseases

Pneumonia is an inflammation of the airspaces (alveoli; singular alveolus) in the lung most commonly caused by infections. Bacteria, viruses, or fungi (infrequently) can cause the infection. There are also a few noninfectious types of pneumonia that are caused by inhaling or aspirating foreign matter or toxic substances into the lungs. Some cases of pneumonia are

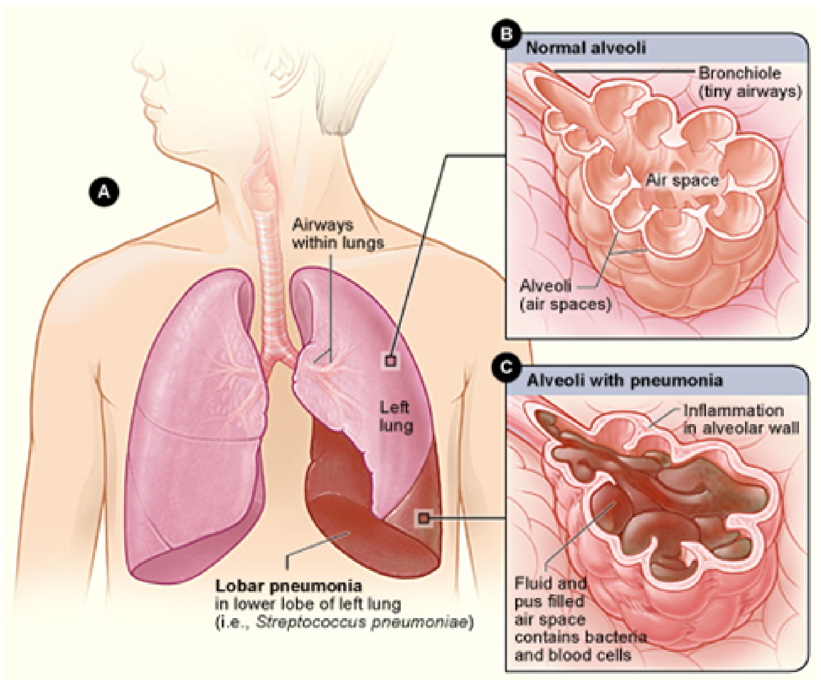
A
Airways within lungs
Left lung
Lobar pneumonia in lower lobe of left lung (i.e., *Streptococcus pneumoniae*)
B
Normal alveoli
Bronchiole (tiny airways)
Air space
Alveoli (air spaces)
C
Alveoli with pneumonia
Inflammation in alveolar wall
Fluid and pus filled air space contains bacteria and blood cells

Fig. 2.1 Pneumonia is an infection that inflames human lungs' air sacs (alveoli)[11]

life-threatening. Around 50,000 people die each year of pneumonia in the U.S. Although anyone of any age can be affected, pneumonia is more common in elderly people and often occurs when the immune system becomes weakened via a prior infection or another condition. Pneumonia is generally more serious when it affects older adults, infants and young children, those with chronic medical conditions, or those with weakened immune function. Most pneumonia occurs when a breakdown in human body's natural defenses allows germs to invade and multiply within lungs. To destroy the attacking organisms, white blood cells rapidly accumulate. Along with bacteria and fungi, they fill the air sacs within lungs (alveoli). Lung diseases and pulmonary abnormalities are one of the leading causes of mortality all over the world. According to the National Center For Health Statistics, nearly 150 thousand people died of lung cancer in the USA in 2016 [Society] and more than 33 million suffer from chronic lung diseases. In 2017, approximately 2.56 million people died of pneumonia including 800,000 children aged less than five years[10]. Moreover, Scientists and physicians can't reduce the overall mortality rate significantly since 1990[32]. The death rate is even higher in underdeveloped countries. In Bangladesh, More than 38,500 people lost their lives due to pneumonia in 2017. Sub-Saharan and Southeast Asia have the highest mortality rate in the world[10]. Again, In 2012, 345 people for every 100,000 had one or more episodes of pneumonia, down from 307 per 100,000 in 2004. In 2009, this rose to 409 people for every 100,000 due to a global flu pandemic. Around 220,000 people receive a diagnosis of pneumonia each year. Some individuals receive more than one diagnosis within a year, but we have focused on the number of individuals who received a diagnosis, rather than the number of cases.

2.3 What causes pneumonia?

Bacteria are the most common cause. Bacterial pneumonia can occur on its own. It can also develop after you've had certain viral infections such as a cold or the flu. Several different types of bacteria can cause pneumonia, including

- Streptococcus pneumoniae
- Legionella pneumophila
- Mycoplasma pneumoniae
- Chlamydia pneumoniae
- Haemophilus influenzae

Viruses that infect the respiratory tract may cause pneumonia. Viral pneumonia is often mild and goes away on its own within a few weeks. But sometimes it is serious enough that you need to get treatment in a hospital. If you have viral pneumonia, you are at risk of also getting bacterial pneumonia. The different viruses that can cause pneumonia include

- Respiratory syncytial virus (RSV)
- Some common cold and flu viruses
- SARS-CoV-2, the virus that causes COVID-19
- Chlamydia pneumoniae
- Haemophilus influenzae

Fungal pneumonia is more common in people who have chronic health problems or weakened immune systems. Some of the types include

- Pneumocystis pneumonia (PCP)
- SARS-CoV-2, the virus that causes COVID-19
- Histoplasmosis
- Cryptococcus

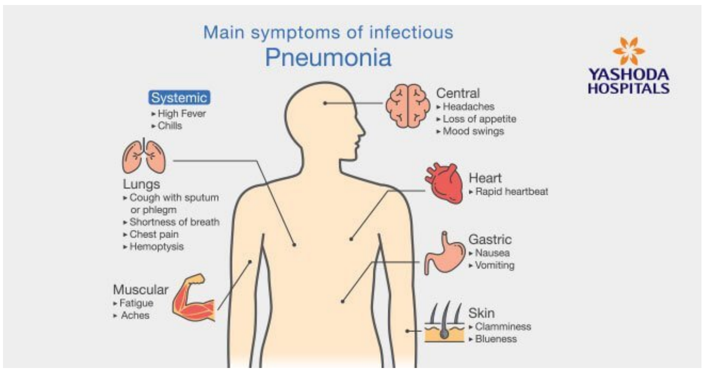

Fig. 2.2 Pneumonia Signs and Symptoms[20]

2.4 Signs ans Symptoms

The signs and symptoms of pneumonia vary from mild to severe, depending on factors such as the type of germ causing the infection, and your age and overall health. Mild signs and symptoms often are similar to those of a cold or flu, but they last longer.

- Cough, which may produce greenish, yellow or even bloody mucus
- Shortness of breath
- Rapid, shallow breathing
- Sharp or stabbing chest pain that gets worse when you breathe deeply or cough
- Loss of appetite, low energy, and fatigue
- Nausea and vomiting, especially in small children
- Confusion, especially in older people
- Sharp or stabbing chest pain (May feel it more when coughing or taking a deep breath)
- Lips and fingernails turning blue
- Fast breathing and heartbeat

2.5 Chest X-ray Interpretation For Pneumonia

Doctors generally ask patients about their medical history and symptoms. Doctors also undergo a physical exam, So. doctor can listen to the patient's lung. In checking for

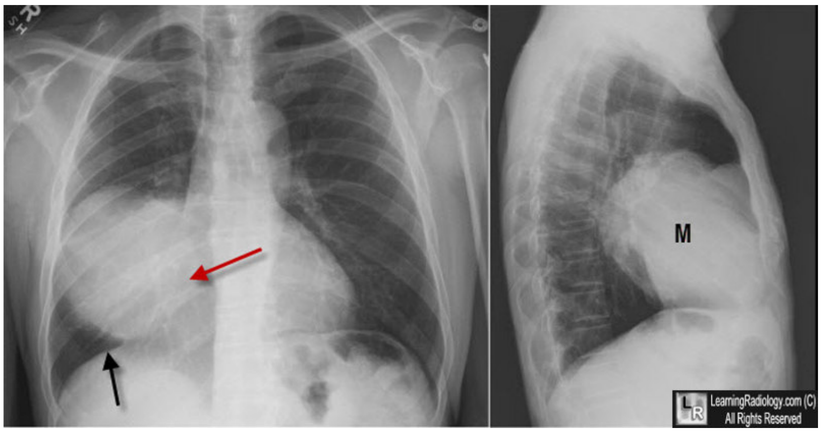

Fig. 2.3 Silhouette Sign, Right Middle Lobe Mass. On the frontal image, there is a large mass in the right lower lung field. We note it is "silhouetting" the right heart border (red arrow) which is no longer seen as a distinct edge. It is not silhouetting the right hemidiaphragm (black arrow). The mass is therefore (1) touching the right heart border and is anterior and (2) the mass is soft tissue or fluid density. The lateral view shows the mass (M) is in the right middle lobe. It was a large bronchogenic carcinoma[26].

pneumonia, doctors will listen for abnormal sounds like crackling, rumbling or wheezing. If the doctor thinks the patient may have pneumonia, an imaging test may be performed to confirm the diagnosis. Pneumonia can best be found on a CXR when a silhouette sign is revealed. A silhouette sign occurs when two structures of equal opacity are next to each other but the border of neither structure can be seen. (See Silhouette sign.) The silhouette sign is sometimes used to distinguish anterior from posterior structures on a CXR. The silhouette sign can help the practitioner determine which lung lobe is affected. However, pneumonia can also present as a localized infiltrate, opacity, or consolidation. These infiltrates can affect any lobe. When attempting to decide which lobe the infiltrate occupies, look for the silhouette sign. When the silhouette sign is seen in the anterior structures, the pneumonia is in the left or right upper lobe of the lung. If the silhouette sign obscures the border of the right or left hemidiaphragm, this indicates lower lobe pneumonia. When the right lower lobe border is lost but the right hemidiaphragm is visible, right middle lobe pneumonia is present.

2.6 Conclusion

In this chapter, The relation between lung opacity and pneumonia have been discussed. So, It can be said that lung opacity refers to pneumonia-related diseases. The fatality of pneumonia and its related disease have also been discussed. We also learned that how pneumonia can be interpreted from chest x-rays.

Chapter 3

Neural Networks (NN)

This chapter discusses different sections of the neural network. Neural network's motivation and its architecture are briefly explained here. The latter part discusses how the model gets to build and how it is learned through the process.

3.1 Neural Networks

Neural networks, a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data. Image classification based on some particular visual information has always been a difficult task even for the human visual system. Considering the chest x-ray images, they are much more difficult to classify because of their complex geometrical shape as well as the inter-intraclass variability in the images. Neural network is a trainable deep learning architecture. This was encouraged by human optical system[27] that learns features directly from the given input data avoiding the hand-crafted feature. It consists of multiple nonlinear transformations in several steps and in each step, it works to reduce the error to accurately classify.

3.2 Biological Motivation

To develop an artificial intelligence(AI), researchers created a highly interconnected system by using the combination of simple computing elements which mimics the working method of a human brain. At first, researchers were trying to imitate the neurophysiology of the brain and these elements acted as the neurons of a brain. As the modern neural networks now are incorporated with different numerical analysis methods, they can make predictions about different real-world problems[6]. The basic computational unit of the brain is a neuron.

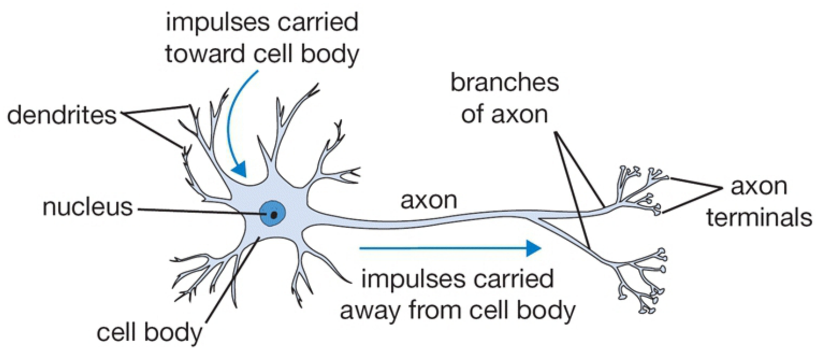
impulses carried toward cell body
dendrites
branches of axon
nucleus
axon
axon terminals
impulses carried away from cell body
cell body

Fig. 3.1 Biological Neuron[6]

Approximately 86 billion neurons can be found in the human nervous system and they are connected with approximately 10^{14}-10^{15}. Figure 3.1 synapses shows a drawing of a biological neuron (top) and a common mathematical model (bottom). Each neuron receives input signals from its dendrites and produces output signals along its (single) axon. The axon eventually branches out and connects via synapses to dendrites of other neurons. In the computational model of a neuron, the signals that travel along the axons (e.g. x0) interact multiplicatively (e.g. w0x0) with the dendrites of the other neuron based on the synaptic strength at that synapse (e.g. w0). The idea is that the synaptic strengths (the weights w) are learnable and control the strength of influence of one neuron on another. In the basic model, the dendrites carry the signal to the cell body where they all get summed. If the final sum is above a certain threshold, the neuron can fire, sending a spike along its axon. In the computational model, we assume that the precise timings of the spikes do not matter and that only the frequency of the firing communicates information. Based on this rate code interpretation, we model the firing rate of the neuron with an activation function f, which represents the frequency of the spikes along the axon. Historically, a common choice of activation function is the sigmoid function σ, since it takes a real-valued input (the signal strength after the sum) and squashes it to range between 0 and 1. So, each neuron performs a dot product with the input and its weights, adds the bias and applies the non-linearity (or activation function).

3.3 Neural Network Architectures

Neural Networks are modeled as collections of neurons that are connected in an acyclic graph. In other words, the outputs of some neurons can become inputs to other neurons. Cycles are not allowed since that would imply an infinite loop in the forward pass of a network. Instead of an amorphous blob of connected neurons, Neural Network models are often organized

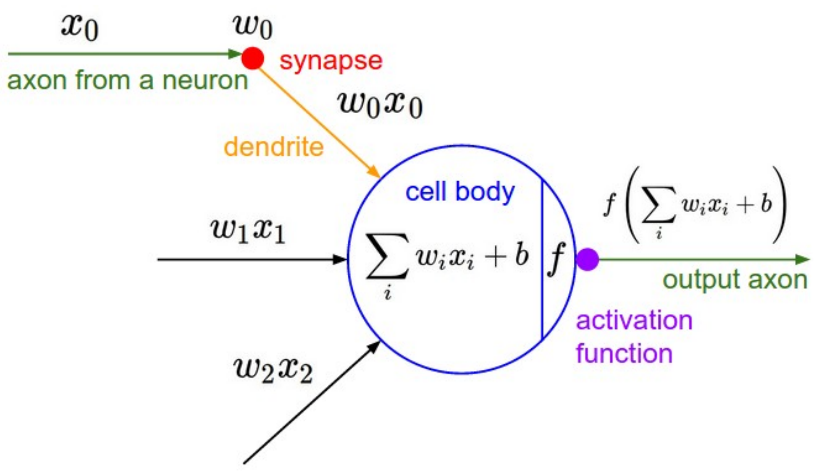

Fig. 3.2 Artificial Neuron[40]

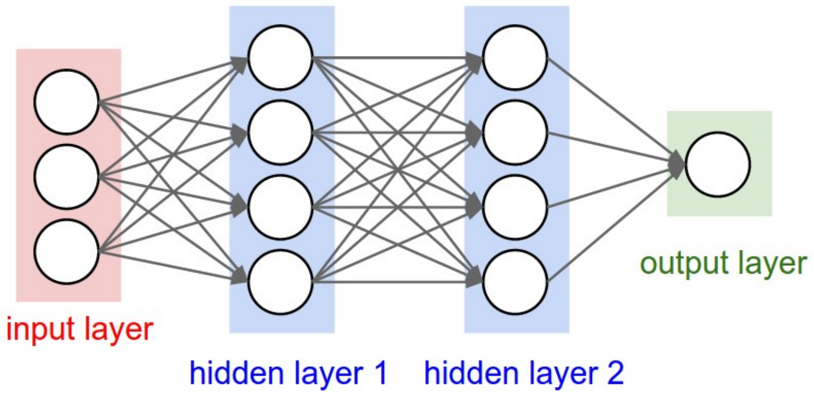

Fig. 3.3 Artificial Neural Network[40]

into distinct layers of neurons. For regular neural networks, the most common layer type is the fully-connected layer in which neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connections. Figure 3.2 shows a 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. In both cases there are connections (synapses) between neurons across layers, but not within a layer.

3.4 Data Preprocessing

Some data preprocessing techniques are used in neural network to involves transforming raw data into an understandable format. Mean subtraction is the most common form of preprocessing. It involves subtracting the mean across every individual feature in the data and has the geometric interpretation of centering the cloud of data around the origin along

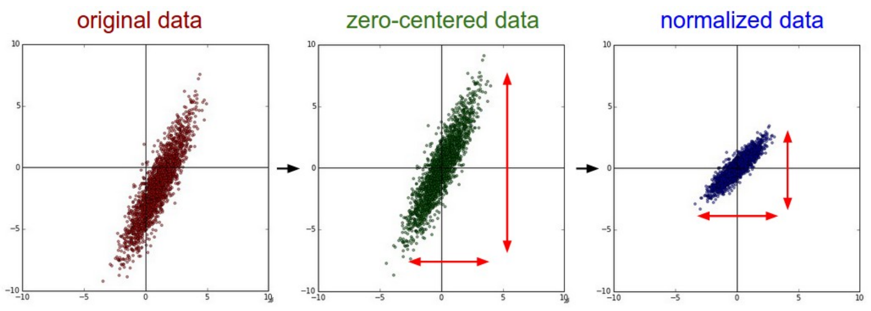

Fig. 3.4 Preprocessing the data[40]

every dimension. Normalization refers to normalizing the data dimensions so that they are of approximately the same scale. There are two common ways of achieving this normalization. One is to divide each dimension by its standard deviation, once it has been zero-centered. Another form of this preprocessing normalizes each dimension so that the min and max along the dimension is -1 and 1 respectively. It is only used when different input features have different scales (or units), but they should be of approximately equal importance to the learning algorithm.

Common data preprocessing pipeline includes the original data, 2-dimensional input data is firstly zero-centered by subtracting the mean in each dimension. The data cloud then gets centered around the origin. Each dimension is additionally scaled by its standard deviation. Figure 3.4 shows this process in three steps. the red lines in the figure indicate the extent of the data.they are of unequal length in the middle, but of equal length on the right.

3.5 Weight Initialization

Before training the network, the convention is to initialize its parameters. A reasonable-sounding idea might be to set all the initial weights to zero, which is the expectation of “best guess” in expectation. But if every neuron in the network computes the same output, then they will also all compute the same gradients during backpropagation and undergo the exact same parameter updates. In other words, there is no source of asymmetry between neurons if their weights are initialized to be the same. As a solution, it is common to initialize the weights of the neurons to small numbers and refer to doing so as symmetry breaking. The idea is that the neurons are all random and unique in the beginning, so they will compute distinct updates and integrate themselves as diverse parts of the full network. With this formulation, every neuron’s weight vector is initialized as a random vector sampled from a multi-dimensional Gaussian, so the neurons point in the random direction in the input space. It is possible and common to initialize the biases to be zero since the asymmetry breaking is provided by the small random numbers in the weights. The Xavier initialization [38] method for weight initialization is an important tool for initializing the weights which are used in this research. Xavier initialization makes sure the weights are appropriate, keeping the signal in a reasonable range of values through many layers. It tries to make sure the distribution of the inputs to each activation function is zero mean and unit variance. To do this, it assumes that the input data has29 been normalized to the same distribution. The greater the number of inputs a neuron has, the smaller the initial weights should be, in order to compensate the number of inputs. In a word, the Xavier initialization method tries to initialize weights with a smarter value, such that neurons won't start training in saturation.

3.6 Regularization

To reduce overfitting some regularization methods are implemented in neural networks. One of them is L2 regularization. L2 regularization is perhaps the most common form of regularization. It can be implemented by penalizing the squared magnitude of all parameters directly in the objective. For every weight w in the network, the term $\frac{1}{2}\lambda\omega^2$ added to the objective, where λ is the regularization strength. It is common to see the factor of $\frac{1}{2}$ in front because then the gradient of this term with respect to the parameter ω is simply $\lambda\omega$ instead of $2\lambda\omega$. The L2 regularization has the intuitive interpretation of heavily penalizing peaky weight vectors and preferring diffuse weight vectors.

L1 regularization is another relatively common form of regularization, where for each weight ω we add the term $\lambda|\omega|$ to the objective. It is possible to combine the L1 regularization with the L2 regularization. The L1 regularization has the intriguing property that it leads the weight vectors to become sparse during optimization (i.e. very close to exactly zero). In other words, neurons with L1 regularization end up using only a sparse subset of their most important inputs and become nearly invariant to the "noisy" inputs. In comparison, final weight vectors from L2 regularization are usually diffuse, small numbers.

Another form of regularization is to enforce an absolute upper bound on the magnitude of the weight vector for every neuron and use projected gradient descent to enforce the constraint. This is known as max norm constraints.

3.7 Backpropagation

When training the network there are mostly two passes which need to be completed one after another. The forward pass computes values from inputs to output and the backward pass then performs backpropagation which starts at the end and recursively applies the chain rule to compute the gradients all the way to the inputs of the circuit. The Backpropagation algorithm is used to learn the weights of a multilayer neural network with a fixed architecture. It performs gradient descent to try to minimize the sum squared error between the network's output values and the given target values. Figure 5.5 explains (depicted as a lined pattern) at the output node and activation (depicted as a solid pattern) at the hidden node. While the change to an input to hidden weight depends on the error at the hidden node (which in turn depend on the error at all the output nodes) and activation at the input node. For the purpose of this derivation:

- The subscript κ denotes the output layer.
- The subscript j denotes the hidden layer.
- The subscript j denotes the hidden layer.
- W_{kj} notes a weight from the hidden to the output layer
- W_{ji} notes a weight from the hidden to the output layer
- a denotes an activation value (sigmoid function).
- t denotes a target value.
- *net* denotes the net input The total error in a network is given by the following equation. All the equations are taken from the study of [40]:

  $$E = \frac{1}{2}\sum(t_k - a_k)^2$$

  We want to adjust the network's weights to reduce this overall error.

  $$\Delta W \propto \frac{\partial E}{\partial W}$$

  We will begin at the output layer with a particular weight.

  $$\Delta W_{kj} \propto \frac{\partial E}{\partial W_{kj}}$$

  However, the error is not directly a function of a weight. We expand this as follows.

  $$\Delta W_{kj} = -\varepsilon \frac{\partial E}{\partial a_k} \frac{\partial a_k}{\partial net_k} \frac{\partial net_k}{\partial W_{kj}}$$

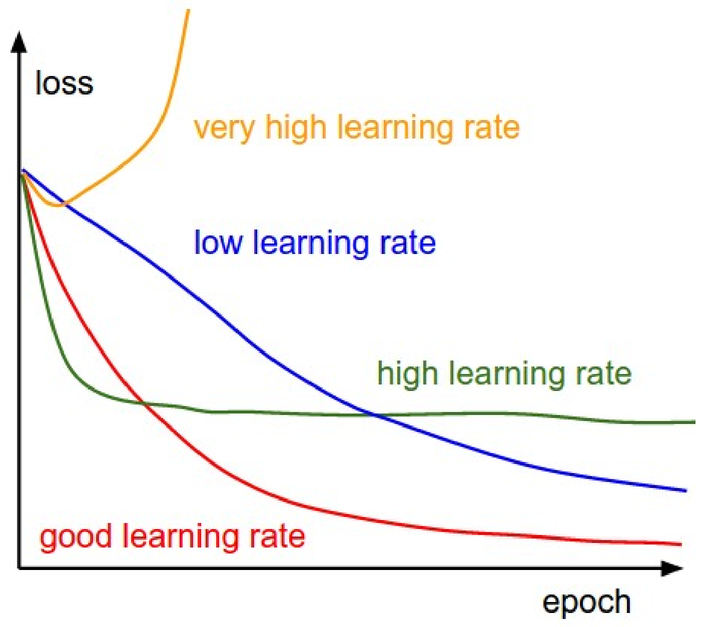

Fig. 3.5 Loss function is changing with different learning rates [40]

Let's consider each of these partial derivatives in turn. Note that only one term of the E summation will have a non-zero derivative: the one associated with the particular weight we are considering. Derivative of the error with respect to the activation:

$$\frac{\partial E}{\partial a_k} = \frac{\partial \left(\frac{1}{2}\sum(t_k - a_k)^2\right)}{\partial a_k} = -(t_k - a_k)$$

3.8 Learning Rate

The rate the weights will be updated is known as the learning rate for the model. The learning rate needs to be controlled as the effects of different learning rates can produce different results and the goal is to minimize the loss function. This is depicted in figure 5.5 as with low learning rates the improvements will be linear. With high learning rates, they will start to look more exponential. Higher learning rates will decay the loss faster, but they get stuck at worse values of loss (green line). This is because there is too much "energy" in the optimization and the parameters are bouncing around chaotically, unable to settle in a nice spot in the optimization landscape.

3.9 Per-Parameter Adaptive Learning Rate Methods

All previous approaches we've discussed so far manipulated the learning rate globally and equally for all parameters. Tuning the learning rates is an expensive process, so much work has gone into devising methods that can adaptively tune the learning rates, and even do so per parameter. Many of these methods may still require other hyperparameter settings, but the argument is that they are well-behaved for a broader range of hyperparameter values than the raw learning rate. In this section, we highlight some common adaptive methods.

Adagrad:

Adagrad is an adaptive learning rate method originally proposed by Duchi et al [12]. In the Adagrad method, the denominator accumulates the squared gradients from each iteration starting at the beginning of training. Since each term is positive, this accumulated sum continues to grow throughout training, effectively shrinking the learning rate on each dimension. In this method, the variable cache has the size equal to the size of the gradient and keeps track of the per-parameter sum of squared gradients. This is then used to normalize the parameter update step, element-wise. The weights that receive high gradients will have their effective learning rate reduced, while weights that receive small or infrequent updates will have their effective learning rate increased. The smoothing term "eps" (usually set somewhere in the range from e$\hat{4}$ to e$\hat{8}$ avoids division by zero. A downside of Adagrad is that in the case of Deep Learning, the monotonic learning rate usually proves too aggressive and stops learning too early.

Adadelta:

Adadelta is a more robust extension of Adagrad that adapts learning rates based on a moving window of gradient updates, instead of accumulating all past gradients [48]. This way, Adadelta continues learning even when many updates have been done. The method dynamically adapts over time using only first order information and has minimal computational overhead beyond vanilla stochastic gradient descent. After many iterations, this learning rate will become infinitesimally small. The method requires no manual tuning of a learning rate and appears robust to noisy gradient information, different model architecture choices, various data modalities, and selection of hyperparameters. The two main drawbacks of Adagrad which were improved in Adadelta:

- The continual decay of learning rates throughout training

- The need for a manually selected global learning rate.

3.10 Conclusion

Neural network tries to mimic the construction of the human optical system. It is generally consisted of three layers. At first, some minimal data processing is done and the weights are initialized of the network. Then the network is ready to train and the gradient is calculated so the weights can be changed in such a way that the loss of the network will be minimized.

Chapter 4

Convolutional Neural Networks (CNN)

This chapter discussed in detail about convolutional neural network. It explores the basic architecture of CNN and how it works to solve image classification problems.

4.1 CNN

Convolutional Neural Networks (CNN) are very similar to the ordinary Neural Network discussed before. CNN is also made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they shave a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer just like an ordinary neural network. The main difference is ConvNet architectures make the explicit assumption that the inputs are images, which allows the model to encode certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the number of parameters in the network. CNN is a trainable deep learning architecture. This was encouraged by animal optical system[27] that learns features directly from the given input data avoiding the handcrafted feature. CNN consists of multiple nonlinear transformations in several steps and in each step, it works to reduce the error to accurately classify. In summary, a CNN is made of multiple trainable stages composed like a stack of data top of each other followed by a supervised classifier. Each stage has a set of arrays named feature vectors which represent the input and output vectors respectively[27]. CNN needs a huge number of labeled samples and computational power to train the network. Because of the growth of the available digital data and powerful computational resources i.e. graphics processing unit (GPU)[19], the required time is being reduced significantly than before while training the network. This convenience can help us to train deeper CNN architectures in our environment to achieve a greater result.

4.2 Architecture of CNN

Before further going to the details, let us explain how a CNN network converges. It is a structured neural network which requires minimal preprocessing where the first a couple of layers are sequentially interconnected in order to process visual information. CNN is feed-forwarding in nature since the output of the current layer becomes the next layer's input. Neurons in the CNN have learnable parameters such as weights and biases. The network starts training itself with a forward pass. A list of connected layers transforms the input volume into an output volume. The prediction in the output layer is computed as the probability that represents class scores. The predicted outcome is then compared with the actual result to compute the error. In backpropagation, the computed error generates the gradient that flows in the backward direction. At each step, the parameters are tuned in such a direction that it tries to reduce the previously generated error[35][19]. This process continues iteratively until the model converges.

Convolutional Neural Networks can take advantage of the fact that the input consists of images and constrains the architecture more sensibly. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) The neurons of a layer will only be connected to a small region of the layer before it, instead of all neurons in a fully-connected manner. A ConvNet is made up of Layers. Every Layer has a simple API: It transforms an input 3D volume to an output 3D volume with some differentiable function that may or may not have parameters. Figure 6.1 depicts this ConvNet process. From the figure, (Top) A regular 3-layer Neural Network. (Bottom) The ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

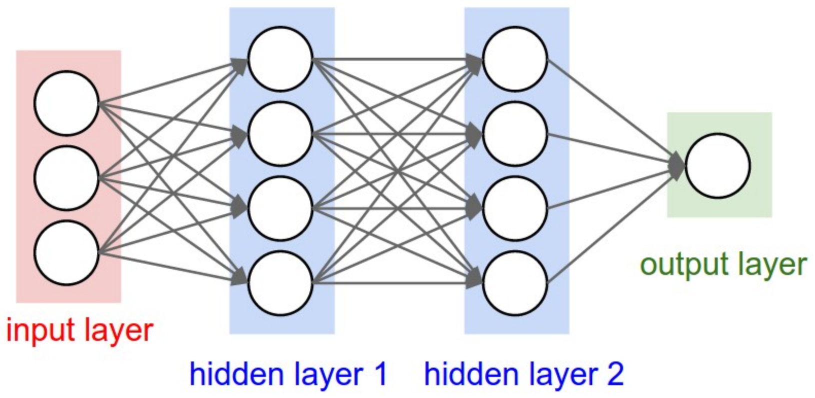

Fig. 4.1 Multilayer Perceptron Neural Networks[40]

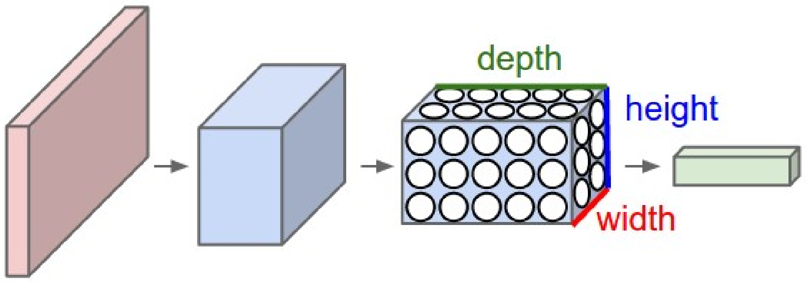

Fig. 4.2 Dimensionality Reduction[40]

4.3 Layers of CNN

A simple ConvNet is sequence of layers, every layer of a ConvNet transforms a volume of activations to another through a differentiable function. Three main types of layers are needed building a ConvNet architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer (precisely as seen in regular Neural Networks). Stack these layers together to form a full ConvNet architecture.

Figure 6.2 shows how a typical CNN architecture looks like: Figure 6.3 describes the

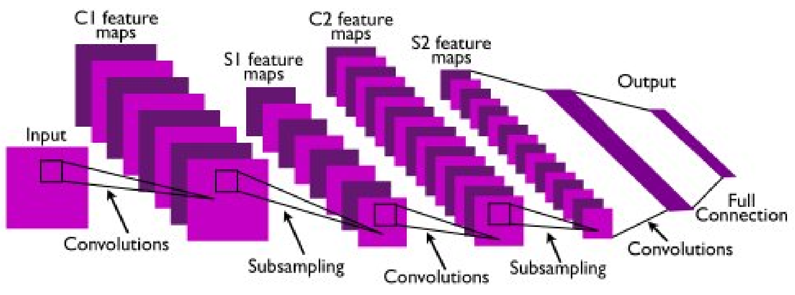

Fig. 4.3 Full Procedure of CNN (Input to Output)[40]

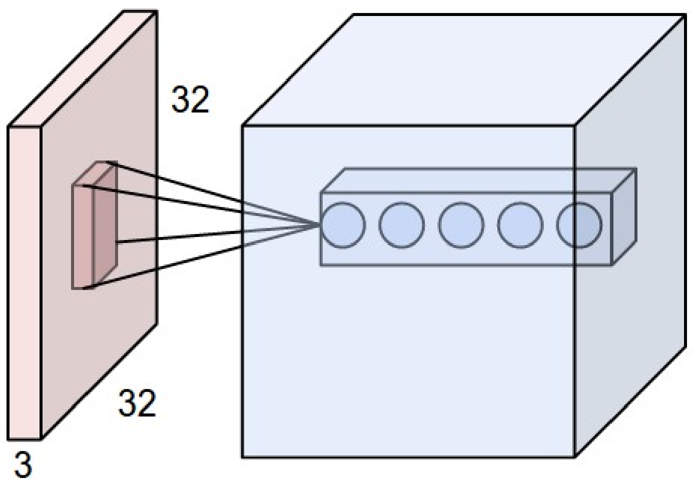

Fig. 4.4 ConV layer to neural network transformation[40]

activations of an example ConvNet architecture. The initial volume stores the raw image pixels (left) and the last volume stores the class scores (right). Each volume of activations along the processing path is shown as a column. Since it's difficult to visualize 3D volumes, we lay out each volume's slices in rows. The last layer volume holds the scores for each class, but here we only visualize the sorted top 5 scores and print the labels of each one. section[Convolutional Layer]Convolutional Layer

The Conv layer is the core building block of Convolutional Network that does most computational heavy lifting. The CONV layer's parameters consist of a set of learnable filters. Every filter is small spatially (along width and height), but extends through the input volume's full depth. For example, a typical filter on the first layer of a ConvNet might have size 5x5x3 (i.e. 5 pixels width and height, and 3 because images have depth 3, the color channels). During the forward pass, each filter is slid i.e. convoluted across the input volume's width and height and compute dot products between the entries of the filter and the input at any position. As we slide the filter over the input volume's width and height, we will produce a 2-dimensional activation map that gives the responses of that filter at every spatial position shown in figure 6.4. Intuitively, the network will learn filters that activate when they see some type of visual features such as an edge of some orientation or a blotch of some color on the first layer or entire honeycomb or wheel-like patterns on higher layers of the network. Now, we will have an entire set of filters in each CONV layer (e.g. 12 filters), and each of them will produce a separate 2-dimensional activation map. We will stack these activation maps along the depth dimension and produce the output volume.

Three hyperparameters control the size of the output volume: the depth, stride, and zero-padding. The depth of the output volume is a hyperparameter: it corresponds to the

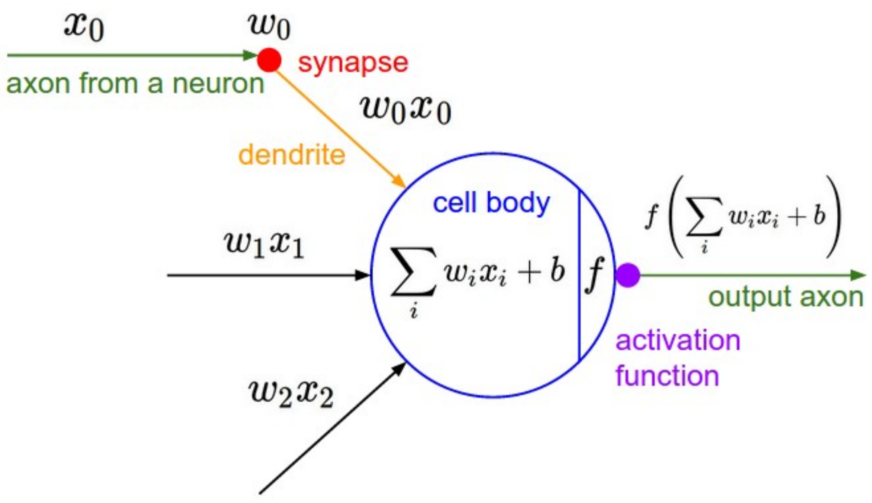

Fig. 4.5 Simple Neural Network[27]

number of filters that are being used to each learning to look for something different in the input. The number of the stride must be specified with which we slide the filter. When the stride is 1 then we move the filters one pixel at a time. When the stride is 2 (or uncommonly 3 or more, though this is rare in practice) then the filters jump 2 pixels at a time as we slide them around. This will produce smaller output volumes spatially. Finally, sometimes it will be convenient to pad the input volume with zeros around the border. The size of this zero-padding is a hyperparameter. The nice feature of zero padding is that it will allow us to control the spatial size of the output volumes.

he spatial size of the output volume can be computed as a function of the input volume size (w), the receptive field size of the Conv Layer neurons (F), the stride with which they are applied (S and the amount of zero padding used (P) on the border. In general, setting zero padding to be $P = \frac{F-1}{2}$ when the stride is S 1 ensures that the input volume and output volume will have the same size spatially.

4.4 Activation Layer

The activation layer introduces non-linear properties to the network followed by convolution layers. Their main purpose is to convert an input signal of a node into an output signal. The generated output will be passed towards the next layer. Every activation function (or non-linearity) takes a single number and performs a certain fixed mathematical operation on it. There are many activation functions to choose from. Two activation functions are discussed here.

- Sigmoid: The sigmoid non-linearity has the mathematical form $\sigma(x) = \dfrac{1}{1+e^{-x}}$ is shown in figure 6.5. It takes a real-valued number and “squashes” it into a range between 0 and 1. In particular, large negative numbers become 0 and large positive numbers become 1. The sigmoid function has seen frequent use historically since it has a nice interpretation as the firing rate of a neuron: from not firing at all (0) to fully-saturated firing at an assumed maximum frequency (1). In practice, the sigmoid non-linearity has recently fallen out of favor and it is rarely ever used. It has two major drawbacks:
  - A very undesirable property of the sigmoid neuron is that when the neuron’s activation saturates at either tail of 0 or 1, the gradient at these regions is almost zero. Therefore, if the local gradient is very small, it will effectively “kill” the gradient and almost no signal will flow through the neuron to its weights and recursively to its data.
  - Sigmoid outputs are not zero-centered. This is undesirable since neurons in later layers of processing in a Neural Network (more on this soon) would be receiving data that is not zero-centered.
- **ReLU:** The Rectified Linear (ReLU) Unit has become very popular in the last few years. It is computationally efficient and converges much faster than most other activation functions [14]. It computes the function $f(x) = max(0,x)$ showed in figure 6.6. In other words, the activation is simply being the threshold at zero (see image above on the left). There are several pros and cons to using the ReLUs:
  - It was found to greatly accelerate the convergence of stochastic gradient descent compared to the sigmoid functions.
  - Compared to sigmoid neurons that involve expensive operations (exponentials, etc.), the ReLU can be implemented by simply thresholding a matrix of activations at zero
  - Unfortunately, ReLU units can be fragile during training and can “die”. For example, a large gradient flowing through a ReLU neuron could cause the weights to update in such a way that the neuron will never activate on any datapoint again. If this happens, then the gradient flowing through the unit will forever be zero from that point on. That is, the ReLU units can irreversibly die during training since they can get knocked off the data manifold.
- Some other activation functions are tanh, maxout, leaky ReLU etc.

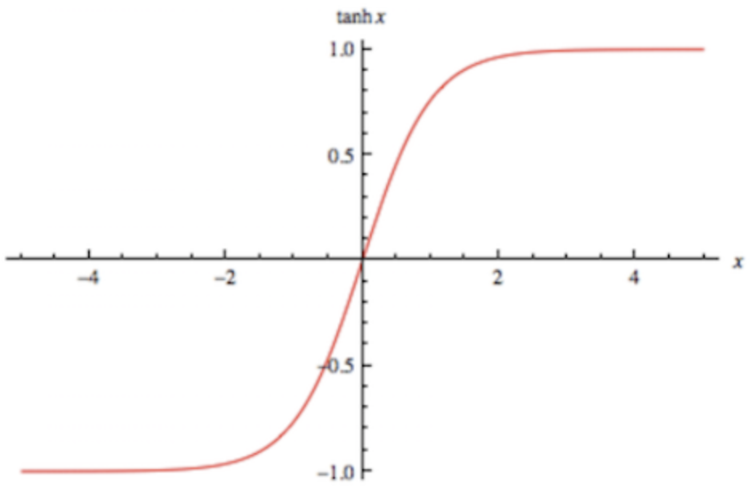

Fig. 4.6 Activation function: tanh[14]

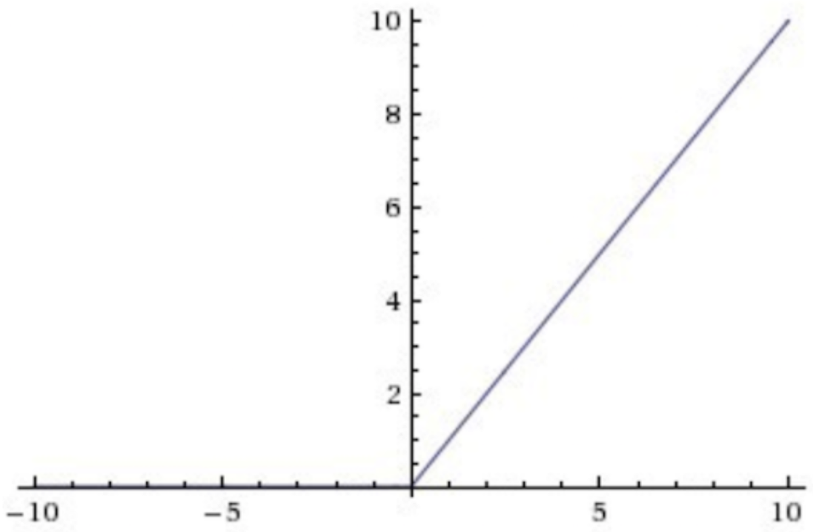

Fig. 4.7 Activation function: ReLU[14]

4.5 Pooling Layer

It is common to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would, in this case, be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged. More generally, the pooling layer:

- Accepts a volume of size $W_1 * H_1 * D_1$
- Requires two hyperparameters:
-
  - their spatial extent F
  - the stride S,
- Produces a volume of size $W_2 * H_2 * D_2$ where:
-
  - $W_2 = \frac{W_1 - F}{S} + 1$
  - $H_2 = \frac{H_1 - F}{S} + 1$
  - $D_2 = D_1$
- Introduces zero parameters since it computes a fixed function of the input
- For Pooling layers, it is not common to pad the input using zero-padding

Pooling layer downsamples the volume spatially, independently in each depth slice of the input volume. In this example of figure 6.7, (top) the input volume of size [224x224x64] is pooled with filter size 2, stride 2 into output volume of size [112x112x64]. Notice that the volume depth is preserved. (Bottom) The most common downsampling operation is max, giving rise to max pooling, here shown with a stride of 2. That is, each max is taken over 4 numbers (little 2x2 square).

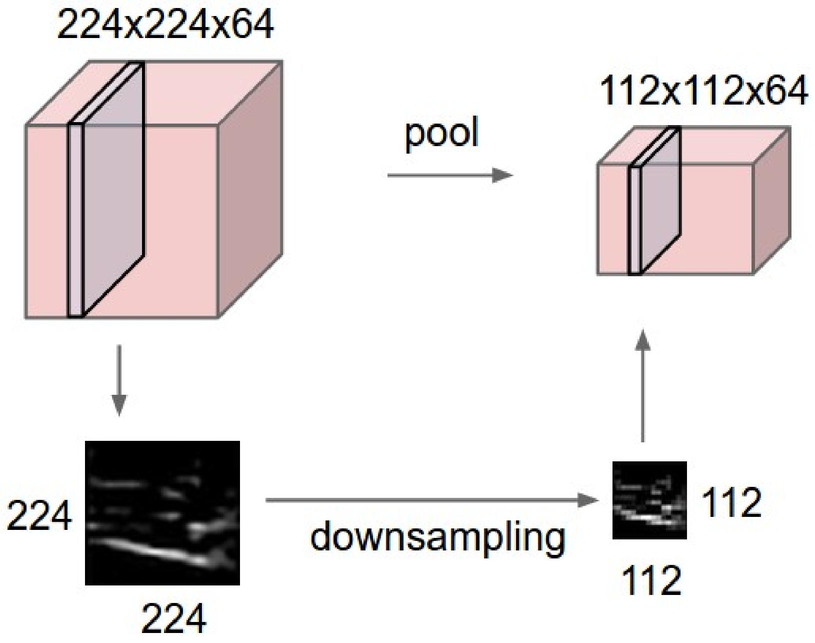

Fig. 4.8 Downsampling by pooling method[40]

4.6 Fully-Connected (FC) Layer

Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset. It is worth noting that the only difference between FC and CONV layers is that the neurons in the CONV layer are connected only to a local region in the input and that many of the neurons in a CONV volume share parameter. However, the neurons in both layers still compute dot products, so their functional form is identical. Therefore, it turns out that it's possible to convert between FC and CONV layers:

- For any CONV layer, there is an FC layer that implements the same forward function. The weight matrix would be a large matrix that is mostly zero except for at certain blocks (due to local connectivity) where the weights in many of the blocks are equal (due to parameter sharing).
- Conversely, any FC layer can be converted to a CONV layer. For example, an FC layer with K=4096 that is looking at some input volume of size 7×7×512 can be equivalently expressed as a CONV layer with F=7,P=0,S=1,K=4096. In other words, we are setting the filter size to be exactly the size of the input volume, and hence the output will simply be 1×1×4096 since only a single depth column "fits" across the input volume, giving an identical result as the initial FC layer. The final fully connected layer of

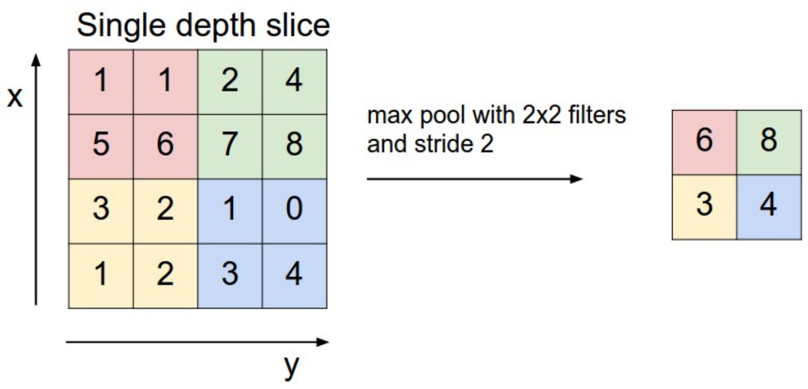

Fig. 4.9 Example of maxpooling[40]

the network produces the net output with an activation function i.e. softmax function depending on the number of classes in the classification problem. A typical CNN architecture consists of these types of several components. An example of a typical architecture would be:

[(CONV-RELU)*N-POOL?]*M-(FC-RELU)*K,SOFTMAX

where N is usually up to 5, M is large, 0 <= K <= 2.

4.7 An Example of CNN Model

A CNN model is described here which can classify child pneumonia from chest X-rays.

The main difference between chest X-ray and other images is chest X-ray images have lung features. Hence, it is important to study the nature of the chest X-ray image to determine the architecture of the proposed model. CNN is a fabulous feature extractor however researchers can optimize it to avoid complex and expensive features[47]. Besides, Larger kernels (i.e., 9x9 or 7x7) may overlook small features and skip essential details where smaller kernels (i.e., 1x1 or 3x3) able to provide more information but it can be confusing for identifying a larger object in images. After considering the above principles, we have proposed a light-weight 6 layers convolutional neural network architecture containing 6.8M parameters.

The First 4 convolutional layers extract lung texture features so they are feature extractors and the other 2 layers are fully connected layers. After trying multiple Kernels, we have achieved the best evaluation metrics result in employing 5x5 size kernel. The input image blocks are extracted by a total of 16 kernels (5x5). We choose the ReLU function in each layer for non-linear activation. Each convolution layer is followed by the Max pooling layers

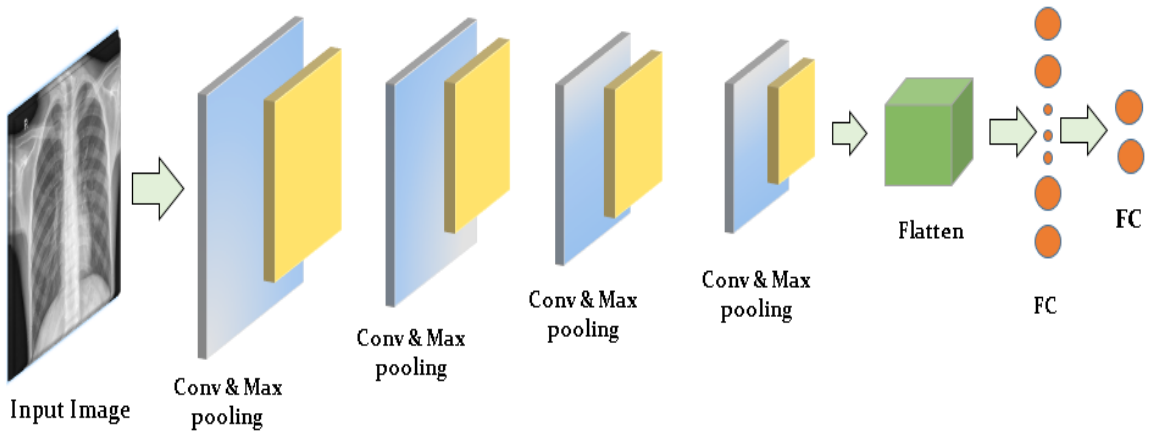

Fig. 4.10 Architecture of proposed convolutional neural network (CNN) model containing 4 convolutional layers and 2 fully connected layers[29]

for extracting the highest pixel values from previous convolutional layer output. Max pooling extracts low-level

features that are very effective for pneumonia image classification along with reducing the computational cost[28]. Output feature map from the 1st convolution layer then extracts by 2nd convolution layer, containing 32 kernels. Likewise, the 3rd and 4th convolution layers extract features by total of 64 and 128 kernels respectively. The resultant feature map then transforms into a 1-D vector. This method is called flattening which is fed to fully connected dense layers for final classification process[41]. Likewise, we feed the flattened vector into 2 fully connected layers.

Hence, The dataset is small, therefore, regularization is applied in neural networks for preventing overfitting. The dropout layer (parameter set to 0.5) is added after flattening for regulization[39]. It stops half of the neurons randomly in each iteration to prevent each unit from being dependent on particular inputs[28]. We have found that the dropout value 0.5 gives the best fit and provides the best training and testing accuracy so far. In the end, the sigmoid activation function is applied at the last dense layer as we perform binary classification. Yet, Choosing the loss function and an algorithm for backpropagation optimization is crucial. An optimization algorithm called Adam is applied to update network weights iteratively which demands less memory requirement instead of traditional stochastic gradient descent procedure. The learning rate (alpha) parameter for Adam optimization is set to 0.001, epsilon is set to 1e-08[29]. The binary cross-entropy method is applied to calculate the loss in each iteration. Further, this architecture was train from scratch without transferring weights from any pre-trained model. The model summery is shown in Table 4.10.

4.8 Our Chosen CNN Architectures

We have chosen 5 architectures to asses the performance.

- **Xception:** Xception is an adaptation from Inception, where the Inception modules have been replaced with depthwise separable convolutions. It has also roughly the same number of parameters as Inception-v1 (23M). Xception takes the Inception hypothesis to an eXtreme (hence the name). Taking this idea to an extreme means performing 1×1 to every channel, then performing a 3×3 to each output. This is identical to replacing the Inception module with depthwise separable convolutions. The hypothesis is [8]:
  - Firstly, cross-channel (or cross-feature map) correlations are captured by 1×1 convolutions.
  - Consequently, spatial correlations within each channel are captured via the regular 3×3 or 5×5 convolutions.
- **DenseNet201:** In DenseNet, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers. Concatenation is used. Each layer is receiving a "collective knowledge" from all preceding layers. DenseNet-201 is a convolutional neural network that is 201 layers deep. Every layer is adding to the previous volume these 32 new feature maps. This is why we go from 64 to 256 after 6 layers. In addition, Transition Block performs as 1x1 convolution with 128 filters. followed by a 2x2 pooling with a stride of 2, resulting on dividing the size of the volume and the number of feature maps on half. We can make new statements from this pattern observation [22].
- **InceptionV3:** Inception-v3 is a convolutional neural network architecture from the Inception family that makes several improvements including using Label Smoothing, Factorized 7 x 7 convolutions, and the use of an auxiliary classifer to propagate label information lower down the network (along with the use of batch normalization for layers in the sidehead). Inception v3 mainly focuses on burning less computational power by modifying the previous Inception architectures. This idea was proposed in the paper [42], published in 2015.
- **MobileNet:** MobileNet is a streamlined architecture that uses depthwise separable convolutions to construct lightweight deep convolutional neural networks and provides an efficient model for mobile and embedded vision applications. Here, Depthwise separable convolution filters are composed of depthwise convolution filters and point convolution filters. The depthwise convolution filter performs a single convolution on

each input channel, and the point convolution filter combines the output of depthwise convolution linearly with 1*1 convolutions [21].

- **ResNet152:** ResNet can have a very deep network of up to 152 layers by learning the residual representation functions instead of learning the signal representation directly. ResNet introduces skip connection (or shortcut connection) to fit the input from the previous layer to the next layer without any modification of the input. Skip connection enables to have deeper network. The authors used the residual block as their network's building block, therefore, during training, when a particular residual block is enable, its input flows through both the identity shortcut and the weight layers, otherwise the input only flows only through the identity shortcut. In training time, each layer has a "survival probability" and is randomly dropped. In testing time, all blocks are kept active and re-calibrated according to its survival probability during training[12] [45] .

4.9 Conclusion

This chapter fully describes how CNN actually works. Convolution, max pooling procedure and activation functions were discussed here briefly. We also described here an example of lightweight CNN architectures. At last, we described briefly the architectures we have chosen.

Chapter 5

Image Augmentation Technique

This chapter discusses data augmentation in image classification problem. It is an important topic for the research as it played a vital role to achieve the calculated accuracy.

5.1 The Necessity of Data Augmentation

Data augmentation has been shown to produce promising ways to increase the accuracy of classification tasks[31]. It is common knowledge that the more data an ML algorithm has access to, the more effective it can be. Even when the data is of lower quality, algorithms can actually perform better, as long as useful data can be extracted by the model from the original data set. For example, text-to-speech and text-based models have improved significantly due to the release of a trillion-word corpus by Google [17]. This result is despite the fact that the data is collected from unfiltered Web pages and contains many errors. With such large and unstructured data sets, however, the task becomes one of finding structure within a sea of unstructured data.

Data augmentation guided by expert knowledge[46], more generic image augmentation, and has shown effective in image classification. Specialized image and video classification tasks often have insufficient data. This is particularly true in the medical industry, where access to data is heavily protected due to privacy concerns. Important tasks such as classifying cancer types are hindered by this lack of data. Techniques have been developed which combine expert domain knowledge with pretrained models. Similarly, small players in the AI industry often lack access to significant amounts of data.

Data augmentation is another way we can reduce overfitting on models, where we increase the amount of training data using the information only in our training data.

Modern deep learning algorithms, such as the convolutional neural network, or CNN, can learn features that are invariant to their location in the image. Nevertheless, augmentation can further aid in this transform invariant approach to learning and can aid the model in learning features that are also invariant to transforms such as left-to-right to top-to-bottom ordering, light levels in photographs, and more.

5.2 Methods

A very generic and accepted current practice for augmenting image data is to perform geometric and color augmentations, such as reflecting the image, cropping and translating the image, and changing the color palette of the image. All of the transformations are affine transformations of the original image that take the form:

$$y = Wx + b$$

The idea has been carried further in[9], where an error rate of 0:35% was achieved by generating new training samples using data augmentation techniques at each layer of a deep network.

5.3 Traditional Transformations

Traditional transformations consist of using a combination of affine transformations to manipulate the training data. For each input image, we generate a “duplicate” image that is shifted, zoomed in/out, rotated, flipped, distorted, or shaded with a hue. Both image and duplicate are fed into the neural net. For a dataset of size N, we generate a dataset of 2N size.

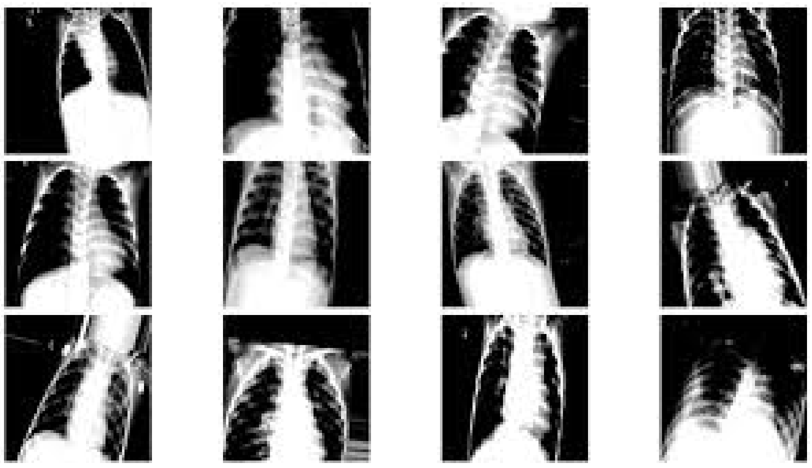

Fig. 5.1 Traditional Transformations[36]

5.4 Learning the Augmentation

During the training phase, there are two parts to the network. The augmentation network takes in two images from the same class as the input image and returns a layer the same size as a single image. This layer is treated as an "augmented" image. The augmented image, as well as the original input image are then passed into the second network, the classification network. The classification loss at the end of the network is a cross entropy loss on the sigmoids of the scores of classes. An addition loss is computed at the end of the augmentation network to regulate how similar the augmented image should be to the input image. The overall loss is a weighted sum of these two losses. We try three different approaches:

- Content loss
- Style loss via gram matrix
- No loss is computed at this layer

5.5 Image Augmentation With Keres *ImageDataGenerator*

The Keras deep learning library provides the ability to use data augmentation automatically when training a model. The class may be instantiated and the configuration for the types of data augmentation are specified by arguments to the class constructor.

A range of techniques are supported, as well as pixel scaling methods. We will focus on five main types of data augmentation techniques for image data[7], specifically:

- Image shifts via the *width_shift_range* and *height_shift_range* arguments.
- Image flips via the *horizontal_flip* and *vertical_flip* arguments.
- Image rotations via the *rotation_range* arguments.
- Image brightness via the *brightness_range* arguments.
- Image zoom via the *zoom_range* arguments.

5.6 Conclusion

Data augmentation technique is pretty useful to achieve better results in classifying objects. Given the plethora of data, we would expect that such data augmentation techniques might be used to benefit not only classification tasks lacking sufficient data, but also help improve the current state of the art algorithms for classification. We applied Keras *ImageDataGenerator* function to augment our training images in training time.

Chapter 6

Result and Performacne Analysis

6.1 Introduction

This chapter consists of the results and performance analysis of the research. It also analyses the data source that we worked with. Later at the chapter, the results are compared with other existing work on this topic.

6.2 Data Source and Description

The dataset was collected from a Kaggle competition named Pneumonia Detection Challenge 2018[30]. It was a two-stage competition and we considered training our models on stage 2 (also containing stage 1 data) dataset. All images of this competition are taken from the NIH database and re-labeled by 6 board-certified radiologists of RSNA (Radiologist Society of North America)[37]. The whole dataset contains a total of 30227 Dicom images (a total of 27227 for training and 3000 for testing) representing three classes.

Lung opacities indicate the presumable lung unhealthy tissue which enervates the X-ray emission so it appears more opaque in chest X-ray[15]. Including, Lung opacities refers to pneumonia-like (pneumonia, infiltration, consolidation) diseases. Moreover, Normal class represents healthy chest X-rays without any pathological finding. "No Lung Opacity / Not Normal” class illustrated pathological findings without any pneumonia-related diseases[37].

Class Name	Total Images in Each Class
Lung Opacity	9555
No Lung Opacity	11821
Normal	8851

Table 6.1 Dataset contains three classes. Inequality between the total number of images per class creates data imbalance which was taken care of by a weighted loss function

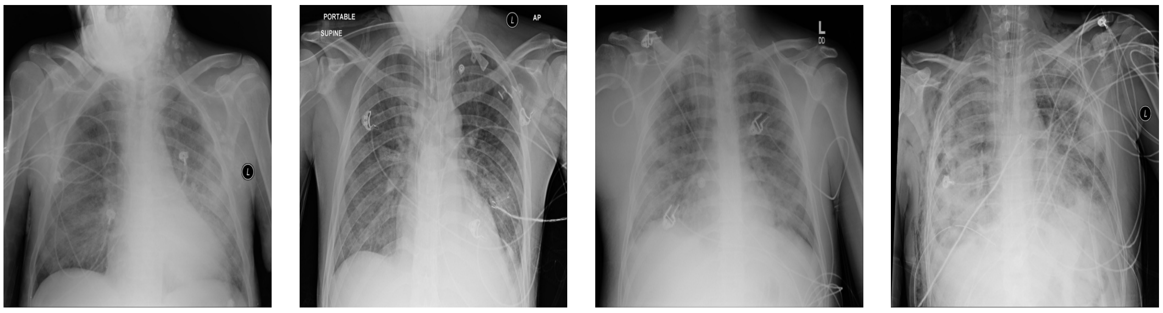

Fig. 6.1 These chest X-rays represent the Lung opacity class from the dataset. Abnormal (more white than normal) regions can be seen[30].

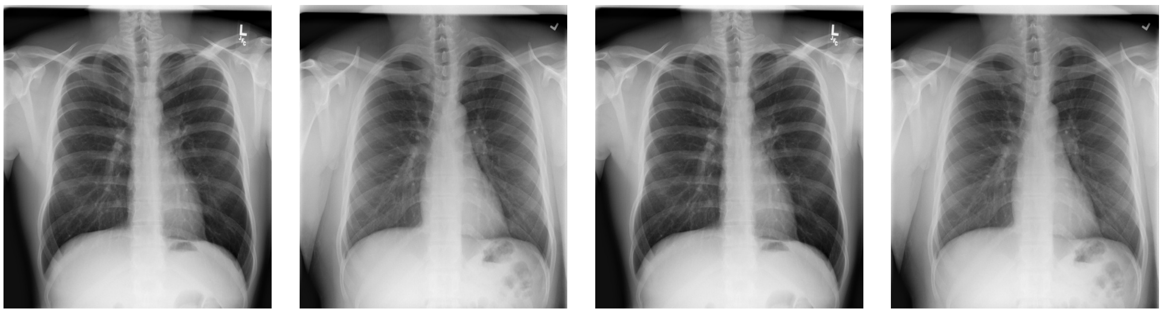

Fig. 6.2 Normal (no findings) chest X-rays[30].

6.3 Weighted Loss Function

It is important to minimize errors for each sample during the training process by a loss function. Consequently, the number of positive and negative samples should be balanced otherwise it creates an imbalance state resulting in biased outcomes from the model. Our specified training set is a class imbalance which is taken care of by weighted binary cross-entropy

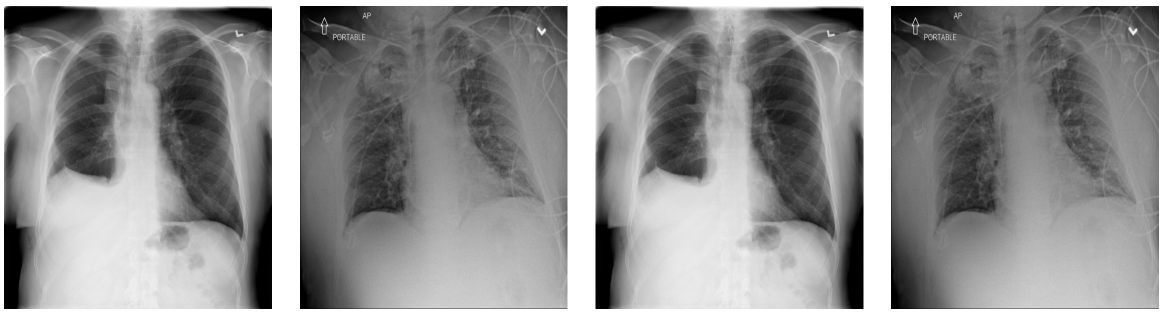
(a) (b) (c) (d)

Fig. 6.3 Abnormal chest X-rays without lung opacity[30].

loss function[34] where $X = \{x_1, x_2, ..., x_n\}$ represents all the input images associating with the ground actual label $y\varepsilon \{0, 1\}$ for all classes.

For a single example in the training set, the following optimized loss function was employed in our study.

$$Lf(X, y) = W_{Pos} \cdot y \log f(x) + W_{Neg} \cdot (1 - y) \cdot \log(1 - f(x))$$

Here, Weighted parameters $W_{Pos} = \frac{1}{N} * T_{Neg}$ and $W_{Neg} = \frac{1}{N} * T_{Pos}$ where T_{Neg} and T_{Pos} refers total number of negative and positive samples in training set respectively. Moreover, N stands for the total number of samples in the training set.

6.4 Evaluation Matrices

The 5 classification matrices, sensitivity/recall, specificity, positive predictive value (PPV)/precision, negative predictive value (NPV) and AUC are important to determine the prediction capability of any model. For binary classification, sensitivity/recall indicates a model's ability to predict true positive cases among all cases that have that disease. Therefore, specificity indicates the opposite, true negatives cases among all cases having no disease in the real case. Along with, positive predictive value (PPV)/precision measures the probability of model positive prediction, If any patient really has the disease[3]. It indicates the classification performance of models over the true positive cases. Conversely, negative predictive value (NPV) measures the probability of model negative prediction, If any patient doesn't have the disease (negative). So, It indicates the classification performance of models over the normal (True negative) cases[29].

$$Sensitivity = \frac{TP}{TP + FN} \tag{6.1}$$

$$Specificity = \frac{TN}{TN + FP} \tag{6.2}$$

$$P = \frac{TP + FN}{TP + TN + FP + FN} \tag{6.3}$$

$$PPV = \frac{Sn * P}{(Sn * P) + (1 - Sp) * (1 - P)} \tag{6.4}$$

$$NPV = \frac{Sp * (1 - P)}{(1 - Sn) * P + Sp * (1 - P)} \tag{6.5}$$

Here, TP, FN, FP, TN, Sn, Sp, P mean true positive, false negative, false positive, true negatives, sensitivity, specificity, and prevalence respectively.

In this research, We compared our model performance by test accuracy, precision/PPV, F1, sensitivity, specificity, NPV and AUC and testing accuracy socres.

6.5 Training Procedure (Both Phase 1 & Phase 2)

CNN models were trained from scratch to find which model performing best to detect lung opacity from chest X-rays. In this study, We divided our training procedure into 2 phases. In phase 1, firstly, fixed training, validation, and test sets were created containing all 3 class images . Then, multiple CNN models were trained on the training set. After that, Models were evaluated on the test set to compare the model's performance. In phase 2, to observe model generalization ability to detect lung opacity from positive and negative (Normal) samples, we excluded the "No Lung Opacity/ Not Normal" class from the whole dataset. Along with, the model performance was validated by the same validation set of respective phases for all models during training.

For each phase, we employed a fixed setup. We trained several well-known deep CNN architectures such as Xception, DenseNet201, Inception V3, MobileNet, ResNet152 from scratch, and assessed the performance of these models with respective test sets of phase 1 & phase 2. All images of the dataset are resized to 224x224 initially. Previously, we also checked model performance with image size 384x384. We observed that there were no notable changes in the evaluation matrices score. Moreover, approximately more than half of training time can be saved if smaller (224x224 resized) images were used for training. Samplewise normalization was employed to the training set wherever validation and test sets were normalized by featurewise normalization. We observed augmented training data increases model generalization ability and prevent overfitting. So, random rotation (maximum 5 degrees) and horizontal flipping were employed in the training stage. Among them, Better AUC scores were achieved by implementing Xception, DenseNet 201, InceptionV3,

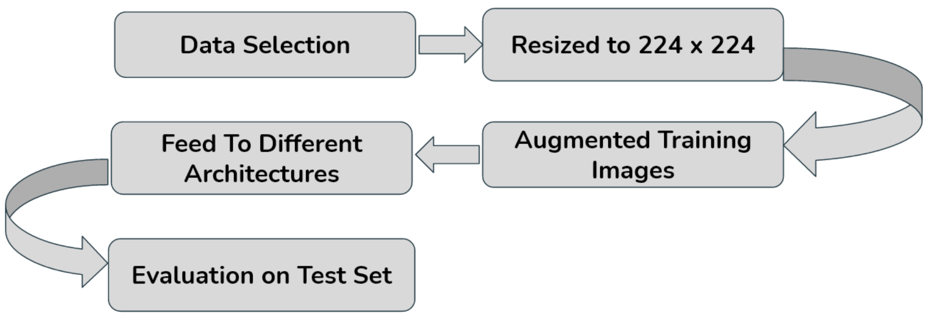

Fig. 6.4 Workks flow of our experiments

ResNet152, and MobileNet separately. No convolutional layer is changed during training. Meanwhile, The last convolutional layer of each model is followed by the global spatial average pooling layer to feed the output to the final fully connected layer. The final fully connected layer of each trained model was replaced by a new fully-connected layer. Moreover, for phase 1, This fully connected layer produces 3-dimensional output for 3 classes after applying the sigmoid function to each output to predict classes but for phase 2, the last fully connected layer was replaced by a new fully-connected layer to produce a binary output.

A weighted binary cross-entropy function (Section **??**) was employed to calculate and update loss in per iteration for both cases. Besides, the Adam optimization algorithm was applied to update network parameters in each iteration. The initial learning rate was set to 0.001 which was decayed by a factor of 10 depending on validation loss if validation loss didn't improve in every 2 epochs. In every epoch, full training data was passed forward and backward through the CNN model once. The total number of the epoch and batch size were set to 16 and 32 respectively. To deal with overfitting, The early stopping method was employed to save weights after every epoch monitoring on validation accuracy. No other methods such as weight decay, batch-normalization, transfer learning, adding dropout were employed[2].

6.6 Phase 1 Training and Result

In Phase 1 , We trained the whole dataset (all three classes) from scratch to see how algorithm will performance over "No Lung Opacity/Not Normal" class. Originally, there are very few difference in lung Opacity and abnormal class.

Class Name	Training	Validation	Test	Total (Per Class)
Lung Opacity	7746	855	954	9555
No Lung Opacity	9581	1034	1206	11821
Normal	7156	832	863	8851
Total	24483	2721	3023	30227

Table 6.2 Dataset was split to train, validation, and test set for phase 1 training and testing. For phase 2, 'No Lung Opacity / Not Normal' class was excluded. The class imbalance was taken care of by a weighted loss function.

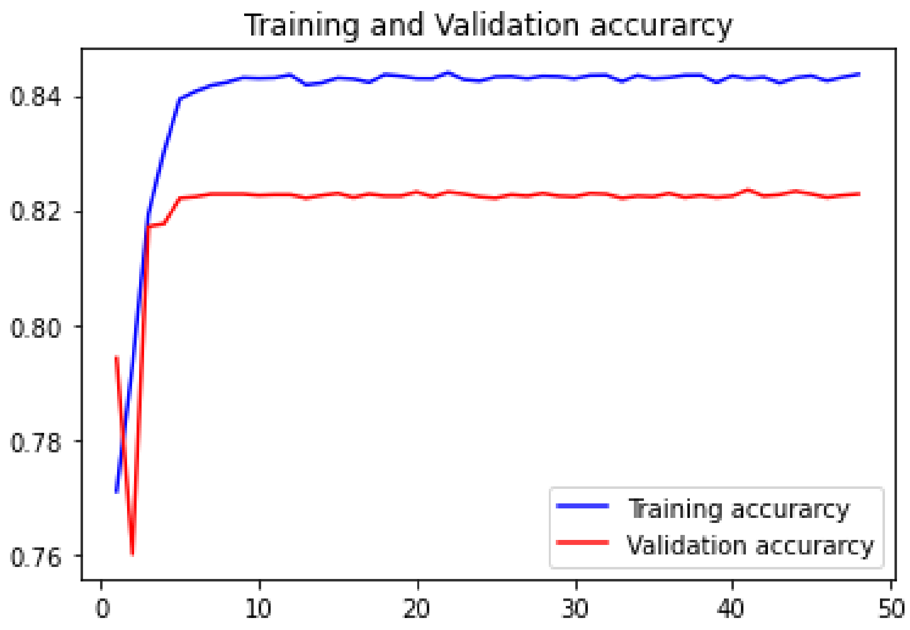

Fig. 6.5 Training and validation accuracy curve during training procedure of Phase 1

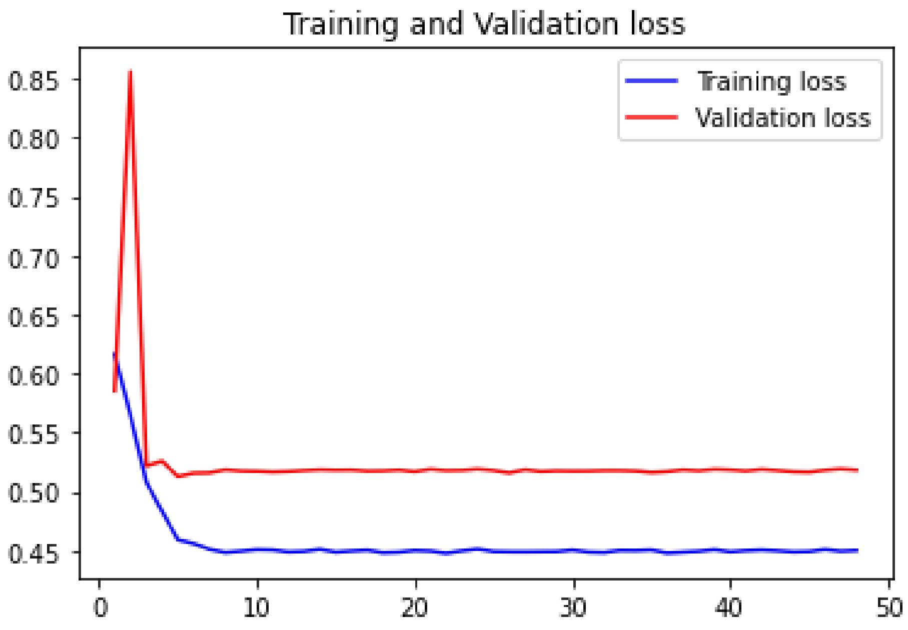

Fig. 6.6 Training and validation loss curve during training procedure of Phase 1

Then, we trained the dataset with each algorithm .

Model	AUC	Sensitivity	Specificity	PPV	NPV	F1	Accuracy
Xception	**91.0**	69.91	90.43	77.10	86.70	73.33	**83.95**
DenseNet201	90.5	**73.48**	88.44	74.57	**87.85**	**74.02**	83.72
InceptionV3	90.3	66.77	**91.44**	**78.25**	85.65	72.05	83.65
MobileNet	90.5	67.40	90.57	76.73	85.76	71.76	83.26
ResNet152	89.5	70.23	88.73	74.20	86.60	72.15	82.9

Table 6.3 Table Shows the performance of top CNN models to classify Lung opacity from chest X-rays over the test set of phase 1 including all 3 classes.

In our research, We evaluated our model performance depends on two types of test sets. Firstly, the training dataset was entirely trained. Then, the first test set containing all three classes was predicted by the models.We have found that Xception achieved the best AUC and accuracy score to predict lung opacity. Meanwhile, In this case, DenseNet201 achieved the best sensitivity, NPV, F1 where InceptionV3 achieved best specificity and PPV score.

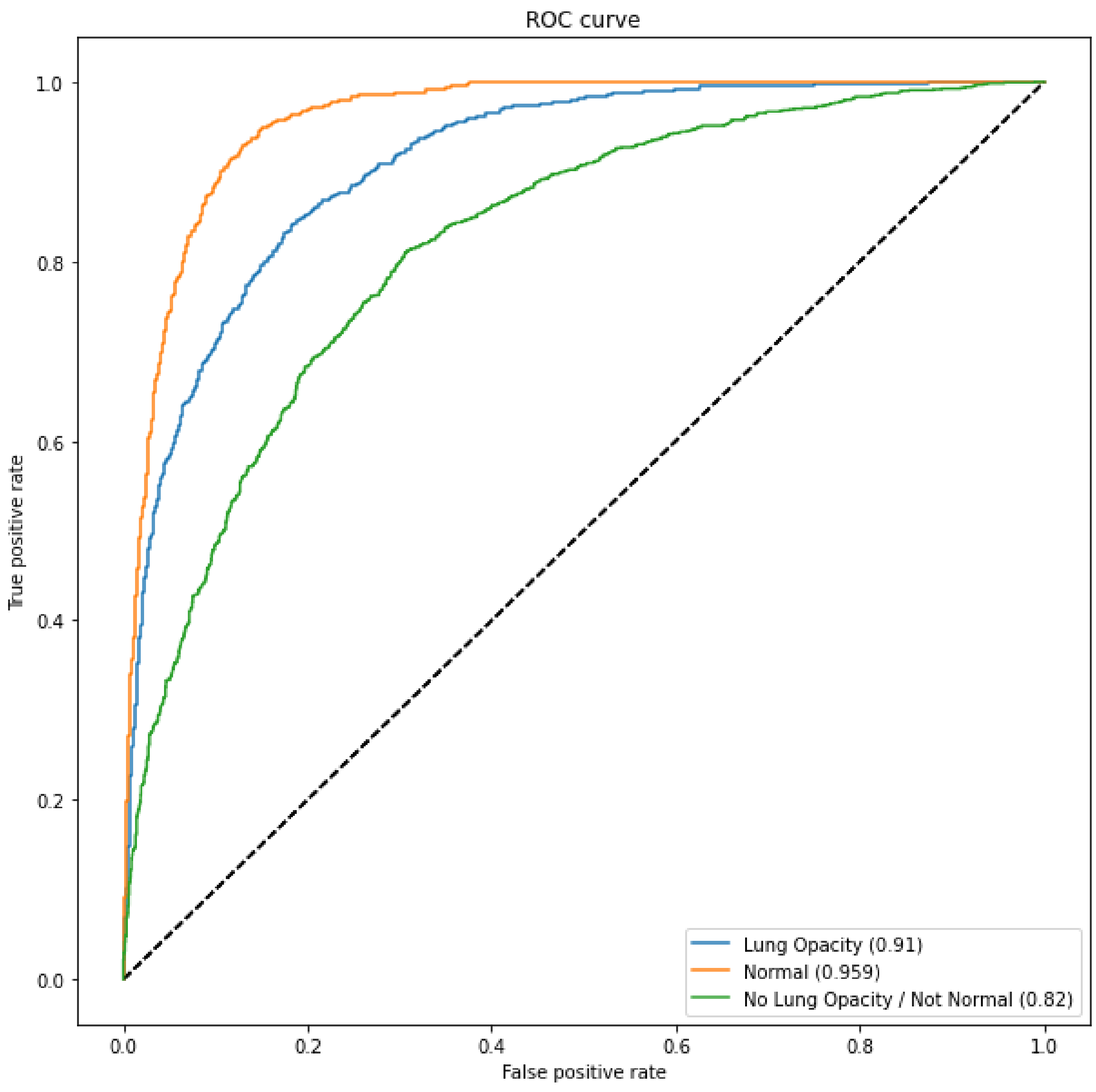

Fig. 6.7 Area Under ROC Curve (AUC) for all classes from Phase 1

6.7 Phase 2 Training and Result

In phase 2, to observe model generalization ability to detect lung opacity from positive and negative (Normal) samples, we excluded the "No Lung Opacity/ Not Normal" class from the whole dataset. Along with, the model performance was validated by the same validation set of respective phases for all models during training.

Class Name	Training	Validation	Test	Total (Per Class)
Lung Opacity	7746	855	954	9555
Normal	7156	832	863	8851
Total	14902	1687	1817	18406

Table 6.4 Dataset was split to train, validation, and test set for phase 1 training and testing. For phase 2, 'No Lung Opacity / Not Normal' class was excluded. The class imbalance was taken care of by a weighted loss function.

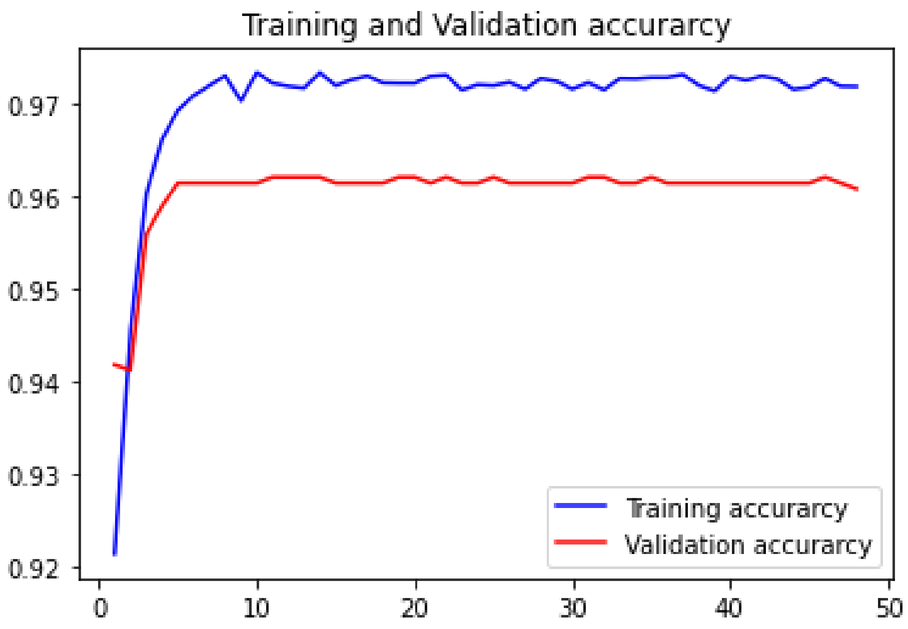

Fig. 6.8 Training and validation accuracy curve during training procedure of Phase 2

Then, we again trained the phase 2 dataset with each algorithm fro scratch.

In this case, Xception outperformed all models considering all performance metrics. We calculated sensitivity and specificity from the study of[43], where Yu-Xing et al. provided a confusion matrix of CNN model performance on the RSNA pneumonia challenge dataset to classify Lung opacity and Normal class. We found that our models achieved better performance metric scores.

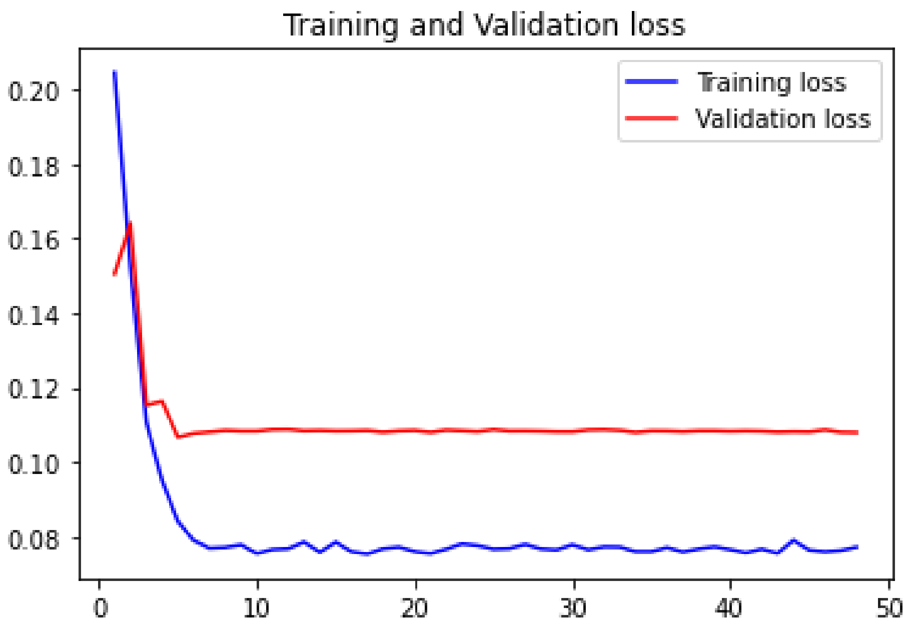

Fig. 6.9 Training and validation loss curve during training procedure of Phase 2

Model	AUC	Sensitivity	Specificity	PPV	NPV	F1	Accuracy
Xception	**99.1**	**97.19**	94.42	93.79	**97.49**	95.46	**95.71**
DenseNet201	98.9	96.96	93.10	92.42	97.24	94.63	94.89
InceptionV3	98.9	96.14	94.11	93.41	96.57	94.75	95.05
MobileNet	**99.1**	95.79	**94.73**	94.02	96.29	94.90	95.22
ResNet152	98.3	93.68	92.80	91.86	94.42	92.76	93.21
Yu-Xing et al[43]	98.04	94.98	94.27	**96.34**	92.20	**95.66**	96.34

Table 6.5 Comparison of performance between top CNN models to classify Lung opacity from Chest X-rays (Lung opacity vs normal classification) over phase 2 test set. Xception achieved the highest AUC and sensitivity..

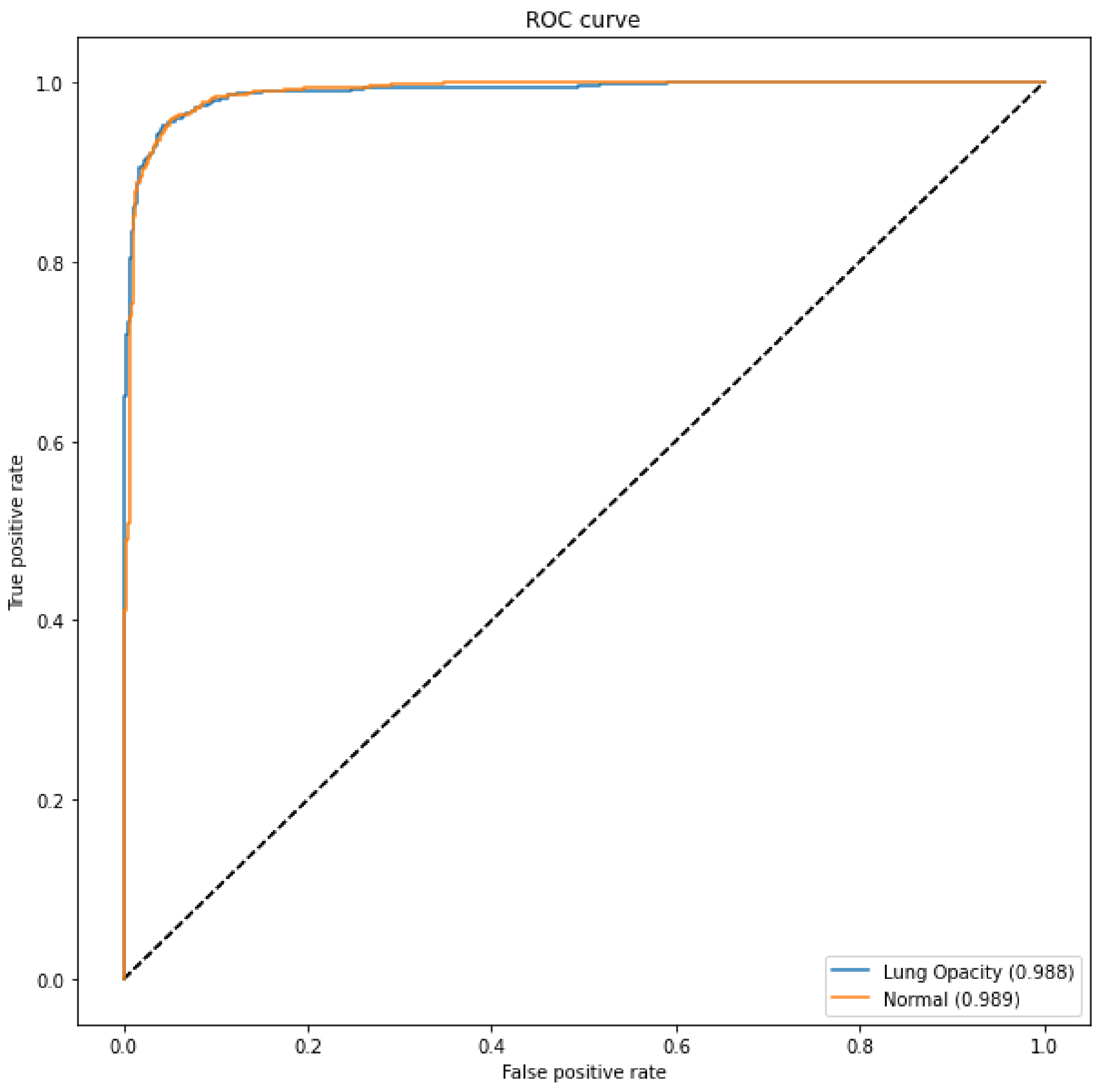

Fig. 6.10 Area Under ROC Curve (AUC) for all classes from Phase 2

6.8 AUC Result For Classifying Other Two Classes

In training phase 2, We calculated the AUC score to classify Lung Opacity from all other classes. We also assessed the AUC score to classify the No Lung Opacity and Normal class. Algorithms have less efficiency to detect No Lung Opacity.

Table 6.6 Comparison of AUC results between different models to detect abnormal and normal findings individually (trained and evaluated in phase 1).

Model	Normal	No Lung Opacity / Not Normal
Xception	**95.9**	**82.0**
DenseNet201	95.1	80.9
Inception V3	95.1	80.2
MobileNet	95.2	80.5
ResNet152V2	94.2	79.2

We see AUC score is dropped at the No Lung Opacity class. The algorithms are confusing to classify this class from other classes (Lung Opacity and Normal Class). We will deal with it in our future research.

6.9 Conclusion

From all the experiments, we can say that Xception architectures showed better performance in the all scenario. "No Lung Opacity/Not Normal" class has similar lung features as other "Lung Opacity" class.

Chapter 7

Conclusion and Future Works

7.1 Overview

In this chapter, We will discuss the summery of the thesis, its limitation and future work direction.

7.2 Summary

In this research, we wanted to assess the performance of CNN models, which can help future researchers build an automatic lung opacity detection system. This CAD reduces the human-made error rate of reviewing chest X-rays. Physicians also can get relief from checking scads of chest X-rays. We trained multiple well-known deep CNN architectures on the RSNA pneumonia challenge dataset to assess the performance of detecting lung opacities from chest X-rays. Xception achieved the best AUC (91.0%) and accuracy (83.95%). Moreover, DenseNet 201 achieved the highest sensitivity (73.48%), NPV (87.85%), and F1 score (74.02%). In these circumstances, Deep CNN architectures failed to achieve higher sensitivity (above 80%). Further studies are necessary to increase sensitivity scores. But, when deep CNN were trained and evaluated on binary (Lung opacity vs Normal Classification) dataset, Xception achieved the highest sensitivity (97.19%), AUC (99.1%), NPV (97.49%), F1 (95.46%), and accuracy (95.71%). We also felt that further research is needed for increasing the classifying performance of "No Lung Opacity/Not Normal". Xception achieved the highest 82.0% AUC, But it is needed to above 90% at least. Further research is necessary to deal with it.

7.3 Limitations

We have only work with the "RSNA Pneumonia Challange Dataset". We didn't consider any other dataset like "NIH Chest X-ray 14". We worked with only one data source to train our model. more data sources, we would have more variant data samples where the model can be more generalized for detecting lung opacity. This research didn't show any guidelines to increase the performance of the "No Lung Opacity/Not Normal" class. No transfer learning procedure was used. Dataset sample labels were not rechecked by any team of physicians.

7.4 Future Work

To work with limitation is to be the first priority future research plan. The total plan includes:

- A new hybrid model architecture which would work to recognize objects much better.
- Extract additional features and fusion with CNN architecture for better classification performance.
- Deal with "No Lung Opacity/Not Normal" class.

7.5 Conclusion

This research is done to help future researchers to choose the right algorithm to build a CAD. In this chapter, We discussed our limitations and the future perspective of our research. The research was done successfully. We want to thank all the researchers of the literature, from whom we took the references. We also thank Google Colab and Kaggle authorities for their notebook and GPU facility.

References

[1] (2020). 140,000 children in bangladesh could die in the next decade unless more is done to fight pneumonia.

[2] Ahamed, M. A., Hossain, M. A., and Al Mamun, M. (2020). Semantic segmentation of self-supervised dataset and medical images using combination of u-net and neural ordinary differential equations. In *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 238–241. IEEE.

[3] Akobeng, A. K. (2007). Understanding diagnostic tests 1: sensitivity, specificity and predictive values. *Acta paediatrica*, 96(3):338–341.

[4] Anitha, V. and Murugavalli, S. (2016). Brain tumour classification using two-tier classifier with adaptive segmentation technique. *IET computer vision*, 10(1):9–17.

[5] Baltruschat, I. M., Nickisch, H., Grass, M., Knopp, T., and Saalbach, A. (2019). Comparison of deep learning approaches for multi-label chest x-ray classification. *Scientific reports*, 9(1):1–10.

[6] Bishop, C. M. et al. (1995). *Neural networks for pattern recognition*. Oxford university press.

[7] Brownlee, J. (2019). How to configure image data augmentation in keras.

[8] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

[9] Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220.

[10] Dadonaite, B. (2018). Pneumonia. *Our World in Data*. https://ourworldindata.org/pneumonia.

[11] Dadonaite, B. and Roser, M. (2018). Pneumonia.

[12] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

[13] Gabruseva, T., Poplavskiy, D., and Kalinin, A. (2020). Deep learning for automatic pneumonia detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 350–351.

[14] Goddard, P., Leslie, A., Jones, A., Wakeley, C., and Kabala, J. (2001). Error in radiology. *The British journal of radiology*, 74(886):949–951.

[15] Goodman, L. R. (2014). *Felson's principles of chest roentgenology, a programmed text*. Elsevier Health Sciences.

[16] Gu, X., Pan, L., Liang, H., and Yang, R. (2018). Classification of bacterial and viral childhood pneumonia using deep learning in chest radiography. In *Proceedings of the 3rd International Conference on Multimedia and Image Processing*, pages 88–93.

[17] Halevy, A., Norvig, P., and Pereira, F. (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12.

[18] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[19] Hey, T. and Trefethen, A. (2003). The data deluge: An e-science perspective. *Grid computing: Making the global infrastructure a reality*, pages 809–824.

[20] Hospitals, Y., on October 28, Y. H., and *, N. (2020). Pneumonia: Types, causes, symptoms, diagnosis and treatment.

[21] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

[22] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

[23] Irvin, J., Rajpurkar, P., Ko, M., Yu, Y., Ciurea-Ilcus, S., Chute, C., Marklund, H., Haghgoo, B., Ball, R., Shpanskaya, K., et al. (2019). Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 590–597.

[24] Jaiswal, A. K., Tiwari, P., Kumar, S., Gupta, D., Khanna, A., and Rodrigues, J. J. (2019). Identifying pneumonia in chest x-rays: A deep learning approach. *Measurement*, 145:511–518.

[25] Kermany, D., Zhang, K., and Goldbaum, M. (2018). Labeled optical coherence tomography (oct) and chest x-ray images for classification. *Mendeley data*, 2.

[26] learningradiology, N. (2020). Silhouette sign.

[27] LeCun, Y., Kavukcuoglu, K., and Farabet, C. (2010). Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE international symposium on circuits and systems*, pages 253–256. IEEE.

[28] Liang, G. and Zheng, L. (2019). A transfer learning method with deep residual network for pediatric pneumonia diagnosis. *Computer methods and programs in biomedicine*, page 104964.

[29] Monowar, K. F., Hassan, M. A. M., and Shin, J. (2020). Lung opacity classification with convolutional neural networks using chest x-rays. In *2020 11th International Conference on Electrical and Computer Engineering (ICECE)*. IEEE.

[30] of North America., R. S. (2018). RSNA Pneumonia Detection Challenge . https://www.kaggle.com/c/rsna-pneumonia-detection-challenge,/.

[31] Perez, L. and Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

[32] Podolsky, S. H. (2005). The changing fate of pneumonia as a public health concern in 20th-century america and beyond. *American Journal of Public Health*, 95(12):2144–2154.

[33] Rajpurkar, P., Irvin, J., Ball, R. L., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C. P., et al. (2018). Deep learning for chest radiograph diagnosis: A retrospective comparison of the chexnext algorithm to practicing radiologists. *PLoS medicine*, 15(11):e1002686.

[34] Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C., Shpanskaya, K., et al. (2017). Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*.

[35] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

[36] Sarkar, A. (2020). Deep learning in healthcare-x-ray imaging (part 5-data augmentation and image normalization).

[37] Shih, G., Wu, C. C., Halabi, S. S., Kohli, M. D., Prevedello, L. M., Cook, T. S., Sharma, A., Amorosa, J. K., Arteaga, V., Galperin-Aizenberg, M., et al. (2019). Augmenting the national institutes of health chest radiograph dataset with expert annotations of possible pneumonia. *Radiology: Artificial Intelligence*, 1(1):e180041.

[Society] Society, A. T. Top 20 Pneumonia Facts—2019. https://www.thoracic.org/patients/patient-resources/resources/top-pneumonia-facts.pdf. Accessed: 2020-09-21.

[39] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

[40] stanford.edu, N. (2017). Cs231n: Convolutional neural networks for visual recognition.

[41] Stephen, O., Sain, M., Maduh, U. J., and Jeong, D.-U. (2019). An efficient deep learning approach to pneumonia classification in healthcare. *Journal of healthcare engineering*, 2019.

[42] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

[43] Tang, Y.-X., Tang, Y.-B., Peng, Y., Yan, K., Bagheri, M., Redd, B. A., Brandon, C. J., Lu, Z., Han, M., Xiao, J., et al. (2020). Automated abnormality classification of chest radiographs using deep convolutional neural networks. *NPJ Digital Medicine*, 3(1):1–8.

[44] Taylor, A. G., Mielke, C., and Mongan, J. (2018). Automated detection of moderate and large pneumothorax on frontal chest x-rays using deep convolutional neural networks: A retrospective study. *PLoS medicine*, 15(11):e1002697.

[45] Wu, Z., Shen, C., and Van Den Hengel, A. (2019). Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133.

[46] Xu, Y., Jia, R., Mou, L., Li, G., Chen, Y., Lu, Y., and Jin, Z. (2016). Improved relation classification by deep recurrent neural networks with data augmentation. *arXiv preprint arXiv:1601.03651*.

[47] Yadav, S. S. and Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big Data*, 6(1):113.

[48] Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.