X-RAYS BASED PNEUMONIA DETECTION USING IBM WATSON STUDIO

Project report submitted to

KRISTU JYOTI COLLEGE OF MANAGEMENT AND TECHNOLOGY

CHANGANACHERRY

In partial fulfilment of the requirements for the award of the

MASTER OF COMPUTER APPLICATION (MCA)

BY

LESLY MATHEW

952017

Under the guidance of

Mrs. TINTU VARGHESE

DEPARTMENT OF COMPUTER APPLICATIONS

KRISTU JYOTI COLLEGE OF MANAGEMENT AND TECHNOLOGY, CHANGANACHERRY

JANUARY 2022.

KRISTU JYOTI COLLEGE OF MANAGEMENT AND TECHNOLOGY

CHANGANACHERRY

DEPARTMENT OF COMPUTER APPLICATION

CERTIFICATE

Certify that the project report entitled "**X-RAYS BASED PNEUMONIA DETECTION USING IBM WATSON STUDIO**" is a bonafide report of the project done by **LESLY MATHEW** (**Register No: 952017**) under our guidance and supervision is submitted in partial fulfilment of the Master of Computer Applications, awarded by Mahatma Gandhi University, Kerala and that no part of this work has been submitted earlier for the award of any other degree.

Rev.Fr. Joshy Cheeramkuzhy CMI **Mr. Roji Thomas**

(Principal) **(HOD)**

Department Seal

Mrs. Tintu Varghese **Mrs. Sunandha Rajagopal**

(Project guide) **(Project Coordinator)**

Submitted for presentation and viva held on……

Evaluator(s) 1. 2

i.

This is to certify that

Lesly Mathew

has completed all courses in the learning path

X-rays Based Pneumonia Detection using IBM Watson

a learning path on smartbridge.skillsnetwork.site

Powered by **IBM Developer Skills Network**

Issued by

The SmartBridge

X-rays Based Pneumonia Detection using IBM Watson consists of the following courses:

- Design Thinking
- DevOps Fundamentals
- Agile Training Module

Rav Ahuja
Program Director
Skills Network, IBM

Issued on:

December 02, 2021

Authenticity of this certificate can be validated by going to:

https://smartbridge.skillsnetwork.site/certificates/7331100a-5e79-4e92-a9c2-cabd37dba8c0

Amarender Katkam
Founder & CEO
TheSmartBridge

DECLARATION

I, **Lesly Mathew (Reg. No: 952017)** hereby declare that the Project Report entitled "**X-RAYS BASED PNEUMONIA DETECTION USING IBM WATSON STUDIO"** submitted to **Mahatma Gandhi University** in partial fulfilment of the requirements for the award of degree of **Master of Computer Applications**, is a record of original work done by me under the guidance at **Mrs. Tintu Varghese,** Assistant Professor, Department of Computer Applications, Kristu Jyoti College of Management and Technology, Changanacherry. All information from other sources has been duly referenced and acknowledged in accordance with the University rules.

Place: Changanacherry Name: LESLY MATHEW

Date: ……

ACKNOWLEDGEMENT

Dreams never turn into reality unless a lot of effort and hard work is put into it. And no effort ever bears fruit in the absence of the support and guidance. First of all, I humbly give all the glory and honour to the almighty God for his eternal love and protection throughout the project work

I would like to thank **Rev. Fr. Joshy Cheeramkuzhy C.M.I**.. Principal. Kristu Jyoti College of Management and Technology, for his encouragement and support throughout the course period. I would like to express my sincere gratitude to the Director of MCA **Dr. O.T.George**, for the support and guidance for completing the project work.

I record my sincere thanks and gratitude to **Mr. Roji Thomas**, Head of the department. Department of Computer Applications, for his valuable advice and encouragement during the course of this project work.

I am extremely grateful to **Mrs. Sunandha Rajagopal**, Assistant Professor. Department of Computer Applications, for the valuable suggestions provided which helped me to overcome the major problems encountered during the completion of this project.

I am profoundly grateful to project guide **Mrs. Tintu Varghese,** Assistant Professor. Department of Computer Applications, for the valuable guidance, help, suggestions and assessment.

I also express my sincere gratitude to **Shivam Shivhare and Shivani Kapoor** for their guidance and support at **SmartInternz, Hyderabad.**

I also extend my heart full thanks to all faculties of the Department of Computer Applications, Kristu Jyoti College of Management and technology for their inspiring and thought-provoking suggestions and constructive criticism for the successful completion of this project. I also express my thanks to all my family and friends for their boundless encouragement and appreciation.

LESLY MATHEW

TABLE OF CONTENTS

ABSTRACT

Artificial intelligence has found its use in various fields during the course of its development, especially in recent years with the enormous increase in available data. Its main task is to assist making better, faster and more reliable decisions. Artificial intelligence, machine learning and deep learning are increasingly finding their application in medicine. This project "X-rays Based Pneumonia Detection using IBM Watson Studio" is a Deep Learning Convolution Neural Network method that is used to process chest X-ray images in order to support the decision-making process in determining the correct diagnosis. This model has the task to help with a classification problem that is detecting whether a chest X-ray shows changes consistent with pneumonia or not, and classifying the X-ray images in two groups depending on the detection results.

The risk of pneumonia is immense for many, especially in developing nations where billions face energy poverty and rely on polluting forms of energy. Over 150 million people get infected with pneumonia on an annual basis especially children below 5 years. A patient suffering from Pneumonia takes an X-ray image to the doctor; with them he predicts pneumonia. The results are not just based on seeing the X-ray images, furthermore, tests will be conducted on the patient. The process was time-consuming, but in recent days artificial intelligence helps in predicting pneumonia bypassing the X-ray image. The main objective of this project is to help the doctors to predict the pneumonia disease more accurately using a deep learning model. The objective is not only to help the doctors but also to the patients to precisely predict pneumonia.

1. INTRODUCTION

CHAPTER-1
INTRODUCTION

1.1 GENERAL INTRODUCTION

Pneumonia is a form of an acute respiratory infection that affects the lungs. The lungs are made up of small sacs called alveoli, which fill with air when a healthy person breathes. When an individual has pneumonia, the alveoli are filled with pus and fluid, which makes breathing painful and limits oxygen intake. Pneumonia is a life-threatening infectious disease affecting one or both lungs in humans commonly caused by bacteria called Streptococcus pneumoniae. One in three deaths in India is caused due to pneumonia as reported by World Health Organization (WHO). Chest X-Rays which are used to diagnose pneumonia need expert radiotherapists for evaluation. Thus, developing an automatic system for detecting pneumonia would be beneficial for treating the disease without any delay particularly in remote areas. Due to the success of deep learning algorithms in analysing medical images, Convolutional Neural Networks (CNNs) have gained much attention for disease classification. In addition, features learned by pre-trained CNN models on large-scale datasets are much useful in image classification tasks. In this project "X-rays Based Pneumonia Detection using IBM Watson Studio", the functionality of pre-trained CNN models utilized as feature-extractors followed by different classifiers for the classification of pneumonic and normal chest X-Rays. The results obtained demonstrates that pretrained CNN models employed along with supervised classifier algorithms can be very beneficial in analyzing chest X-ray images, specifically to detect Pneumonia.

1.2 ORGANIZATIONAL PROFILE

SmartInternz is an experiential learning & externship platform to build the next-gen talent pool. SmartInternz is an Experiential Learning & Remote Externship Platform to bring academia & industry very close for a common goal of talent creation. They are building the Nex-Gen Talent pool with skills in emerging technologies i.e. Artificial Intelligence, Data Science, Internet of Things(loT), Robotics, Blockchain, Quantum Computing and Cyber Security. Our unique models of project based learning, micro-skilling and Internships helps students in building their competency & get ready for industry. They bring the students, educators and employers on a common platform to fill the gap between academia & industry.

SMARTBRIDGE is an edTech organization with a vision to bridge the gap between academia & industry. Our outcome-based experiential learning programs on emerging technologies (Internet of Things, Machine Learning, Data Science, Artificial Intelligence, Robotics) are building skilled entry - level engineers, for the corporate world. SmartBridge is in mission to build technology communities in academia to encourage students towards innovation & entrepreneurship. Since inception, we have trained thousands of students, faculty and working professionals on emerging technologies via technical bootcamps, hackathons, Summer & Winter Internship Programs.

Following are their Services for Academia & Students

- • Technology Bootcamps
- • Hackathons
- **• Hands-on Training Programs**
- • Professional Development Programs
- • Summer Practice Schools

Their main objective is to bridge the existing gaps between prevailing industry standards and what the academics offer to the graduates while passing out of university. SmartBridge offers suitable skill deployment and training to the young talent before on boarding their first job. Their skill development programs are designed considering the present expectations in the industry. We thereby work along the lines to offer best programs in IoT Hyderabad.

Main objectives of SmartBridge

- • Well directed career guidance programs for educational institutions
- • Appropriate certification courses that suit the industry need
- • Train the trainers; expanded awareness about the current industry standards
- • Liaise with corporates to offer niche internships
- • Establish technology development centers in colleges
- • Specialised incubation centers in collaboration with corporate

SMARTBRIDGE EDUCATIONAL SERVICES PRIVATE LIMITED has been created with ingenuity to bridge the existing gaps in the transition phase of young graduates to working

professionals. Their team works at continuous identification of the changing needs in the corporate sector with respect to the talent ramp up and acquisition.

Ever since the inception, we have shown high end captive results to many young graduates in achieving their goals. Their end goals are simple. Fresh graduates should be adequately equipped in terms of skill and confidence before setting sails to their dream careers with the corporates to achieve their Dreams.

Their ever evolving strategies go in sync with what the corporates are really looking at with new talent. We empower the young workforce with the same.

1.3 OBJECTIVES

The risk of pneumonia is immense for many, especially in developing nations where billions face energy poverty and rely on polluting forms of energy. Over 150 million people get infected with pneumonia on an annual basis especially children below 5 years. A patient suffering from Pneumonia takes an X-ray image to the doctor; with them he predicts pneumonia. The results are not just based on seeing the X-ray images, furthermore, tests will be conducted on the patient. The process was time-consuming, but in recent days artificial intelligence helps in predicting pneumonia bypassing the X-ray image. The main objective of the project "X-rays Based Pneumonia Detection using IBM Watson Studio" is to help the doctors to predict the pneumonia disease more accurately using a deep learning model. The objective is not only to help the doctors but also to the patients to precisely predict pneumonia.

1.4 SCOPE AND RELEVANCE OF THE PROJECT

Artificial intelligence has found its use in various fields during the course of its development, especially in recent years with the enormous increase in available data. Its main task is to assist making better, faster and more reliable decisions. Artificial intelligence, machine learning and deep learning are increasingly finding their application in medicine. This is especially true for medical fields that utilize various types of biomedical images and where diagnostic procedures rely on collecting and processing a large number of digital images. The application of deep learning in processing of medical images helps with consistency and boosts accuracy in reporting. This project "X-rays Based Pneumonia Detection using IBM Watson Studio" describes the use of deep learning convolutional neural network to process chest X-ray images in order to support the decision-making process in determining the correct diagnosis. This

model has the task to help with a classification problem that is detecting whether a chest X-ray shows changes consistent with pneumonia or not, and classifying the X-ray images in two groups depending on the detection results.

2. SYSTEM STUDY AND ANALYSIS

CHAPTER 2
SYSTEM ANALYSIS

2.1 INTRODUCTION

System Analysis is the process of gathering and interpreting facts, diagnosing the problems and using the information to recommend improvements. System study is a general term that refers to an orderly, structured process for identifying and solving problems. The first phase of software development is system study.

The importance of system study phase is the establishment of the requirements for the system to be acquired, developed and installed. Analysing the project to understand the complexity forms the vital part of the system study. Problematic areas are identified and information is collected. Fact finding or gathering is essential to any analysis of requirements. It is also highly essential that the analyst familiarize himself with the objectives, activities and functions of organizations in which the system is to be implemented.

In system study, a detailed study of these operations performed by a system and their relationships within and outside the system is done. A key question considered here is, “What must be done to solve the problem?” One aspect of system study is defining the boundaries of the application and determining whether or not the candidate application should be considered.

2.2 EXISTING SYSTEM

In the existing system most of the operations are done manually. Much paper works were required in the existing system. The existing system is not giving accurate results while doing diagnosis. In existing system your doctor will start by asking about your medical history and doing a physical exam, including listening to your lungs with a stethoscope to check for abnormal bubbling or crackling sounds that suggest pneumonia.

If pneumonia is suspected, your doctor may recommend the following tests:

- **Blood tests.** Blood tests are used to confirm an infection and to try to identify the type of organism causing the infection. However, precise identification isn't always possible.

- **Chest X-ray.** A patient suffering from Pneumonia takes an X-ray image to the doctor; with them he predicts pneumonia. The results are not just based on seeing the X-ray images, furthermore, tests will be conducted on the patient. The process was time-consuming

- **Pulse oximetry.** This measures the oxygen level in your blood. Pneumonia can prevent your lungs from moving enough oxygen into your bloodstream.

- **Sputum test.** A sample of fluid from your lungs (sputum) is taken after a deep cough and analyzed to help pinpoint the cause of the infection.

To avoid all these limitations and make the working more accurately the system needs to be designed with more user involvement

.

2.3 PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The existing system has several disadvantages and many more difficulties to work well. The proposed system tries to eliminate the difficulties and it will be less time consuming. The proposed system will help the doctors to reduce the manual workload and start the treatment of patient fast and effectively. Here Convolutional neural networks (CNNs) is an accurate method to help the doctors to predict whether the patient is affected with pneumonia or not. CNNs is divided into two important parts, feature extraction layer (convolutional layer and pooling layer) and fully connected layer. CNNs method is commonly used for image data classification. Therefore, CNNs is suitable to classify pneumonia based on lung X-ray in order to obtain accurate prediction results. Based on accuracy rate of CNN, it shows that CNNs can be applied to image data (especially lung X-ray) for classification of pneumonia disease.

2.4 FEASIBILITY STUDY

A feasibility analysis evaluates the candidate systems and determines the best system that needs performance requirements. The purpose of feasibility study is to investigate the present system, evaluate the possible application of computer-based methods, select a tentative system,

evaluate the cost and effectiveness of the proposed system, evaluate the impact of proposed system on existing system and as certain the need for new system. Feasibility is carried out to see if the system is technically, economical and operationally feasible.

All projects are feasible when given unlimited resources and infinite time. It is both necessary and prudent to evaluate the feasibility of the project at the earliest possible time.. An estimate is made of whether the identified user may be satisfied using current hardware and software technologies. The study will decide if the proposed system will be cost effective from the business point of view and if it can be developed in the existing budgetary constraints.

The objective of a feasibility study is to test the technical, social and economic feasibility of developing a computer system. This is done by investigating the existing system and generating ideas about a new system. The computer system must be evaluated from a technical viewpoint first, and if technically feasible, their impact on the organization and the staff must be accessed. If a compatible, social and technical system can be devised, then it must be tested for economic feasibility.

There are eight steps involved in a feasibility study,

1. Form a project team and appoint a project leader
2. Prepare system flow chart.
3. Enumerate potential candidate system.
4. Describe and identify characteristics of candidate system.
5. Determine and evaluate performance and cost effectiveness of each candidate system.
6. Weight the system performance and cost.
7. Select the best candidate system.
8. Report project directive management.

2.4.1 Operational Feasibility

Operational feasibility is connected with human organizational and political aspects. The issues considered are the job changes that will be brought about, the organizational structures that will be distributed and the new skills that will be required. Methods of processing and presentation are all according to the needs of clients since they can meet all user requirements here. The proposed system will not cause any problem, any circumstances and will work according to the specifications mentioned. Hence the proposed system is operationally feasible. People are inherently resistant to change and computer has been known to facilitate changes. The system operation is the longest phase in the development life cycle of a system. So operational

feasibility should be given much importance. This system has a user friendly interface. Thus, it is easy to handle.

2.4.2 Technical Feasibility

Technical feasibility is the most important of all types of feasibility analysis. Technical feasibility deals with hardware as well as software requirements. An idea from the outline design to system requirements in terms of input/output files and procedures is drawn and types of hardware and software and the methods required for running the system are analyzed. Technical study is a study of hardware and software requirement. All the technical issue related to the proposed system is dealed during feasibility stage of preliminary investigation produced the following results: While considering the problems of existing systems it is sufficient to implement the new system. The proposed system can be implemented to solve issues in the existing system. It includes the evaluation of how it meets the proposed system. The assessment of technical feasibility must be based on the outline of the system requirements in terms of inputs, outputs, files, programs and procedures. This can be quantified in terms of volumes of data, trends, frequency of updating, etc.

2.4.3 Economic Feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of software, more commonly known as the cost/benefit analysis. The procedure is to determine the benefits and savings that are expected from a candidate and compare them with costs. If the benefits outweigh cost, the decision is made to design and implement the system; otherwise further alternatives have to be made. Here it is seen that no new hardware or software is needed for the development of the system.

2.4.4 Behavioural Feasibility

Behavioural feasibility determines how much effort will go in to educating, selling and training the user on the candidate system. People are inherently resistant to change and computers have been known to facilitate change. Since the system is user friendly, user training is a very easy matter.

2.4.5 Legal Feasibility

Legal feasibility is the determination of any infringement, violation, or liability that could result from the development of the system. Legal feasibility environment passes abroad range of concerns that include contract and liability. The proposed project is also a legally feasible one.

2.5 ARTIFICIAL INTELLIGENCE

The intelligence demonstrated by machines is known as Artificial Intelligence. Artificial Intelligence has grown to be very popular in today's world. It is the simulation of natural intelligence in machines that are programmed to learn and mimic the actions of humans. These machines are able to learn with experience and perform human-like tasks. As technologies such as AI continue to grow, they will have a great impact on our quality of life. It's but natural that everyone today wants to connect with AI technology somehow, may it be as an end-user or pursuing a career in Artificial Intelligence. Artificial Intelligence is the broader family consisting of Machine Learning and Deep Learning as its components.

3 Types of Artificial Intelligence

- Artificial Narrow Intelligence (ANI)
- Artificial General Intelligence (AGI)
- Artificial Super Intelligence (ASI)

What is Artificial Narrow Intelligence (ANI)?

These Artificial Intelligence systems are designed to solve one single problem and would be able to execute a single task really well. By definition, they have narrow capabilities, like recommending a product for an e-commerce user or predicting the weather. This is the only kind of Artificial Intelligence that exists today. They're able to come close to human functioning in very specific contexts, and even surpass them in many instances, but only excelling in very controlled environments with a limited set of parameters.

What is Artificial General Intelligence (AGI)?

AGI is still a theoretical concept. It's defined as AI which has a human-level of cognitive function, across a wide variety of domains such as language processing, image processing, computational functioning and reasoning and so on. An AGI system would need to comprise of thousands of Artificial Narrow Intelligence systems working in tandem, communicating with each other to mimic human reasoning.

What is Artificial Super Intelligence (ASI)?

ASI is seen as the logical progression from AGI. An Artificial Super Intelligence (ASI) system would be able to surpass all human capabilities. This would include decision making, taking rational decisions, and even includes things like making better art and building emotional relationships.

2.6 MACHINE LEARNING

Machine learning is a subsidiary of artificial intelligence that facilitates a techniques where machine can make decision based on its experience and improve and learn with time and use without explicitly programmed. Machine learning focuses on the development of computer programs that can access the data and use it to learn for themselves. Machine learning algorithms are often categorized as supervised and unsupervised machine learning algorithms.

2.6.1 Supervised Learning

In supervised learning, the machine is taught by example. The operator provides the machine learning algorithm with a known dataset that includes desired inputs and outputs, and the algorithm must find a method to determine how to arrive at those inputs and outputs. While the operator knows the correct answers to the problem, the algorithm identifies patterns in data, learns from observations and makes predictions. The algorithm makes predictions and is corrected by the operator – and this process continues until the algorithm achieves a high level of accuracy/performance.

Under the umbrella of supervised learning fall: Classification, Regression and Forecasting.

1. **Classification**: In classification tasks, the machine learning program must draw a conclusion from observed values and determine to what category new observations belong. For example, when filtering emails as 'spam' or 'not spam', the program must look at existing observational data and filter the emails accordingly.
2. **Regression**: In regression tasks, the machine learning program must estimate – and understand – the relationships among variables. Regression analysis focuses on one dependent variable and a series of other changing variables – making it particularly useful for prediction and forecasting.
3. **Forecasting**: Forecasting is the process of making predictions about the future based on the past and present data, and is commonly used to analyse trends.

2.6.2 Unsupervised Learning

Unsupervised learning is the type of machine learning algorithm where there is no any defined or labelled class and it itself draws the inferences from datasets. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. Under the umbrella of unsupervised learning, fall:

1. **Clustering**: Clustering involves grouping sets of similar data (based on defined criteria). It's useful for segmenting data into several groups and performing analysis on each data set to find patterns.
2. **Dimension reduction**: Dimension reduction reduces the number of variables being considered to find the exact information required.

2.6.3 Reinforcement Learning

Reinforcement learning focuses on regimented learning processes, where a machine learning algorithm is provided with a set of actions, parameters and end values. By defining the rules, the machine learning algorithm then tries to explore different options and possibilities, monitoring and evaluating each result to determine which one is optimal. Reinforcement learning teaches the machine trial and error. It learns from past experiences and begins to adapt its approach in response to the situation to achieve the best possible result.

2.7 DEEP LEARNING

This is because deep learning models are capable of learning to focus on the right features by themselves, requiring little guidance from the programmer. Basically, deep learning mimics the way our brain functions i.e. it learns from experience. As you know, our brain is made up of billions of neurons that allows us to do amazing things. Actually, our brain has sub-consciously trained itself to do such things over the years. Now, the question comes, how deep learning mimics the functionality of a brain? Well, deep learning uses the concept of artificial neurons that functions in a similar manner as the biological neurons present in our brain. Therefore, we can say that Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.

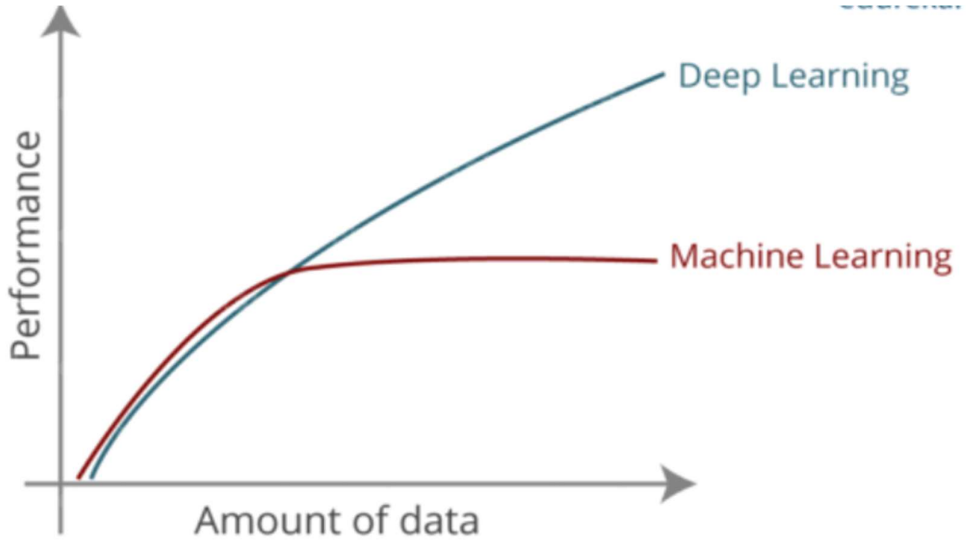


Fig 1: Graph to show performance of ML and DL for given data

2.7.1 Artificial Neural Network:

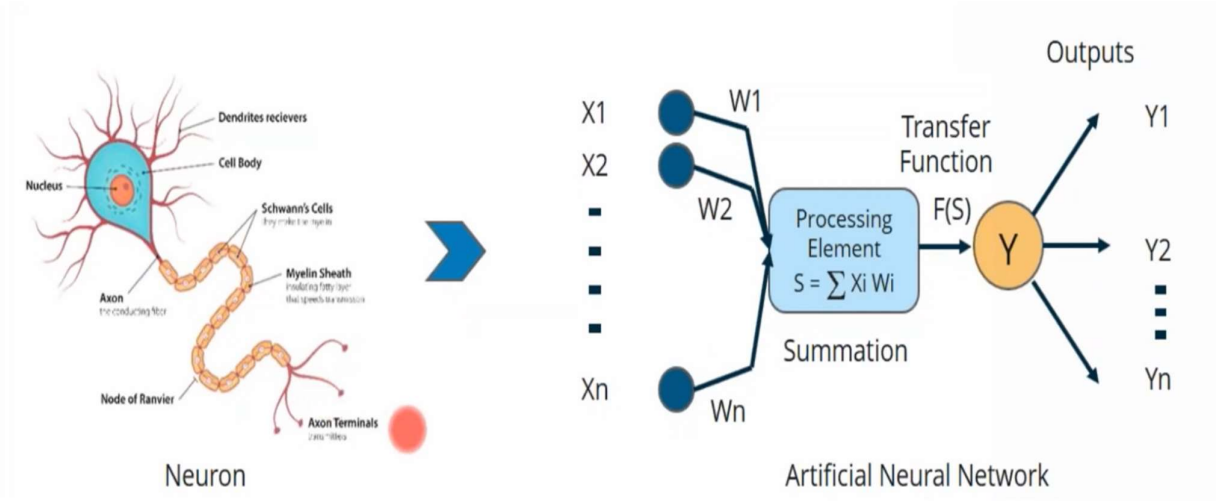


Fig 2: Human neuron and artificial neural network

The deep learning is implemented with help of neural networks and the idea of motivation behind the networks are nothing but neurons. What are neurons? Neurons (also called neurones or nerve cells) are the fundamental units of the brain and nervous system, the cells responsible for receiving sensory input from the external world, for sending motor commands to our muscles, and for transforming and relaying the electrical signals at every step in between. A useful analogy is to think of a neuron as a tree. A neuron has three main parts: dendrites, an axon, and a cell body or soma (see image below), which can be represented as the branches, roots and trunk of a tree, respectively. A dendrite (tree branch) is where a neuron receives input from other cells. Dendrites are used to provide input to neuron there are multiple dendrites, so these many inputs will be provided to neuron. Dendrites branch as they move towards their tips, just like tree branches do, and they even have leaf-like structures on them called spines. The soma is the cell body where the nucleus lies which perform some function, where the neuron's DNA is housed, and where proteins are made to be transported throughout the axon and dendrites. The axon (tree roots) ,the output is travelled through axon and it will go towards the axon on terminals and then the neuron will fire the output to the next neuron, it sends an electrical message called an action potential throughout the entire axon. Two neurons are never connected to each other. There is a gap between them called synopsis. This is how basically a neuron works. On the right side, figure of artificial neural network is similar to neuron with multiple inputs. It has 4 important components:

1. Inputs
2. Weights and Bias
3. Summation Function
4. Activation or transformation Function

- *Inputs*

The inputs (x) received from the input layer are multiplied with their assigned weights w. The multiplied values are then added to form the Weighted Sum. The weighted sum of the inputs and their respective weights are then applied to a relevant Activation Function. The activation function maps the input to the respective output.

- Weights and Bias

Once an input variable is fed to the network, a randomly chosen value is assigned as the weight of that input. The weight of each input data point indicates how important that input is in predicting the outcome. The bias parameter, on the other hand, allows you to adjust the activation function curve in such a way that a precise output is achieved.

- Summation Function

Once the inputs are assigned some weight, the product of the respective input and weight is taken. Adding all these products gives us the Weighted Sum. This is done by the summation function.

- Activation Function

The main aim of the activation functions is to map the weighted sum to the output. Activation functions such as tanh, sigmoid and so on are examples of transformation functions. The activation function compares the actual output and obtained output. If the obtained output is not same then it finds the difference between the actual output and desired output and backpropagate to update the weights. It keeps on updating until we get the desired output.

A neural network consists of three important layers:

- **Input Layer:** As the name suggests, this layer accepts all the inputs provided by the programmer.

- **Hidden Layer:** Between the input and the output layer is a set of layers known as Hidden layers. In this layer, computations are performed which result in the output.
- **Output Layer:** The inputs go through a series of transformations via the hidden layer which finally results in the output that is delivered via this layer.

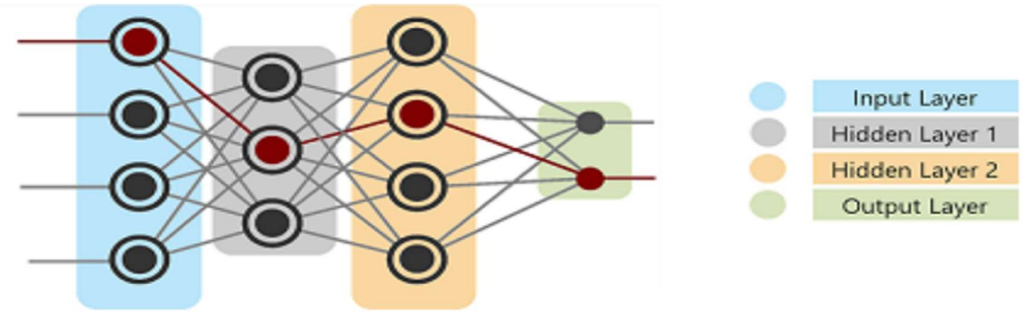


Fig 3: Artificial neural network layer

2.8 CONVOLUTIONAL NEURAL NETWORKS (ConvNets OR CNNs)

Convolutional neural networks (ConvNets or CNNs) are more often utilized for classification and computer vision tasks. Prior to CNNs, manual, time-consuming feature extraction methods were used to identify objects in images. However, convolutional neural networks now provide a more scalable approach to image classification and object recognition tasks, leveraging principles from linear algebra, specifically matrix multiplication, to identify patterns within an image. That said, they can be computationally demanding, requiring graphical processing units (GPUs) to train models.

How do convolutional neural networks work?

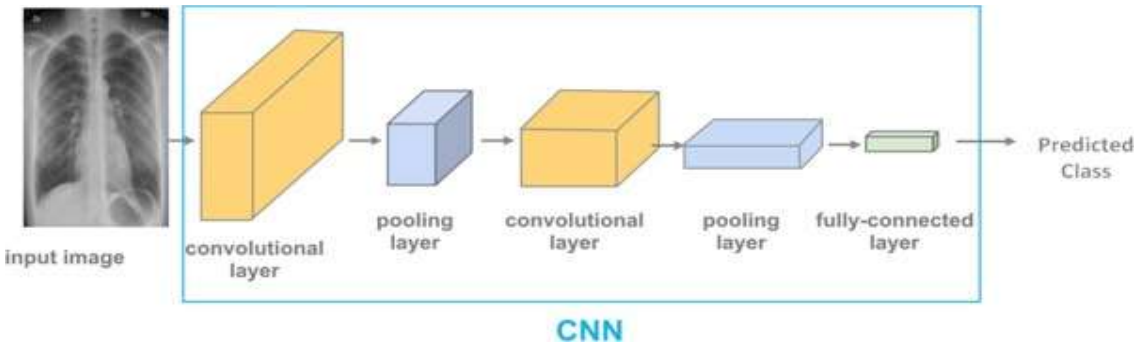


Fig 4: General overview of CNN working

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

2.8.1 Convolutional Layer

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let's assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.

The feature detector is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3x3 matrix; this also determines the size of the receptive field. The filter is then applied to an area of the image, and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has swept across the entire image. The final output from the series of dot products from the input and the filter is known as a feature map, activation map, or a convolved feature.

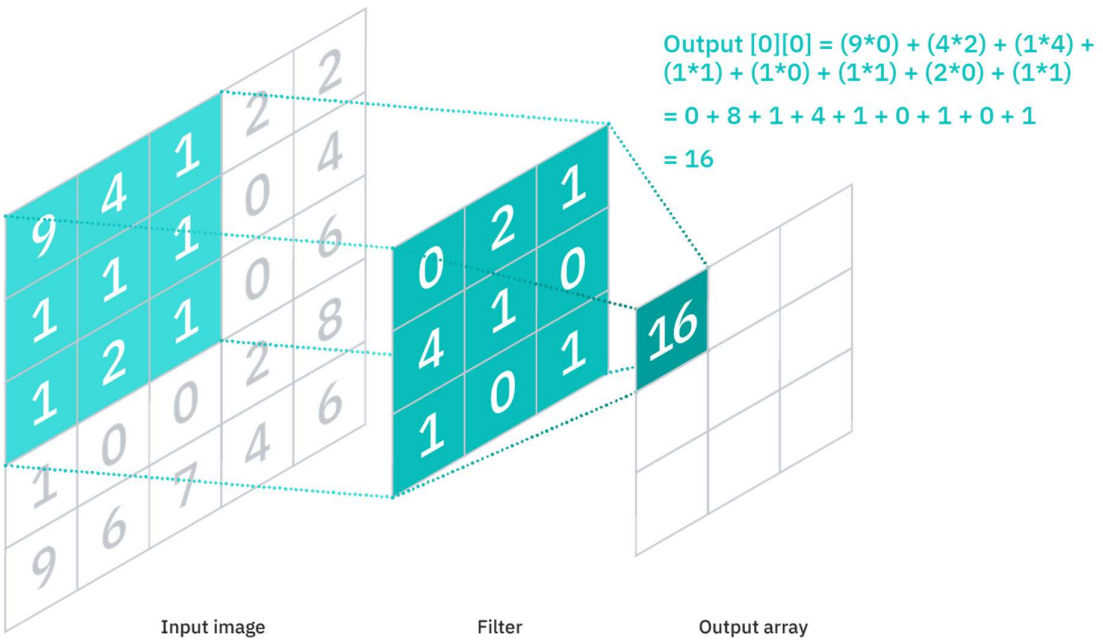
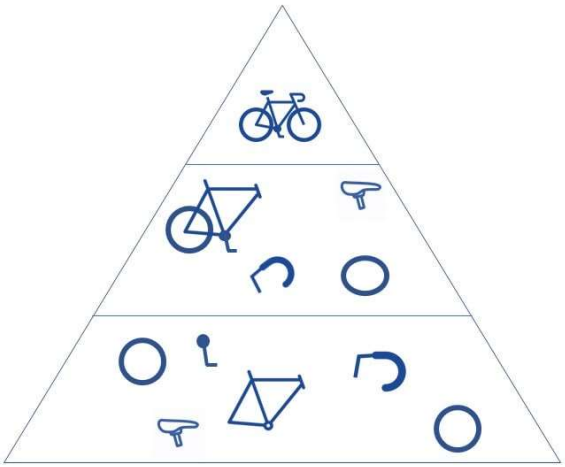


Fig 5: Applying filter to input data in CNN

As you can see in the image above, each output value in the feature map does not have to connect to each pixel value in the input image. It only needs to connect to the receptive field, where the filter is being applied. Since the output array does not need to map directly to each input value, convolutional (and pooling) layers are commonly referred to as "partially connected" layers. However, this characteristic can also be described as local connectivity.

Note that the weights in the feature detector remain fixed as it moves across the image, which is also known as parameter sharing. Some parameters, like the weight values, adjust during training through the process of backpropagation and gradient descent.

As we mentioned earlier, another convolution layer can follow the initial convolution layer. When this happens, the structure of the CNN can become hierarchical as the later layers can see the pixels within the receptive fields of prior layers. As an example, let's assume that we're trying to determine if an image contains a bicycle. You can think of the bicycle as a sum of parts. It is comprised of a frame, handlebars, wheels, pedals, et cetera. Each individual part of the bicycle makes up a lower-level pattern in the neural net, and the combination of its parts represents a higher-level pattern, creating a feature hierarchy within the CNN.



Ultimately, the convolutional layer converts the image into numerical values, allowing the neural network to interpret and extract relevant patterns.

2.8.2 Pooling Layer

Pooling layers, also known as downsampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

- **Max pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
- **Average pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency.

2.8.3 Fully-Connected Layer

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.

This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1.

2.9 LIFE CYCLE OF CNN MODEL

The general life cycle of CNN model is represented in the following figure 7. It encompasses with various stages described below:-

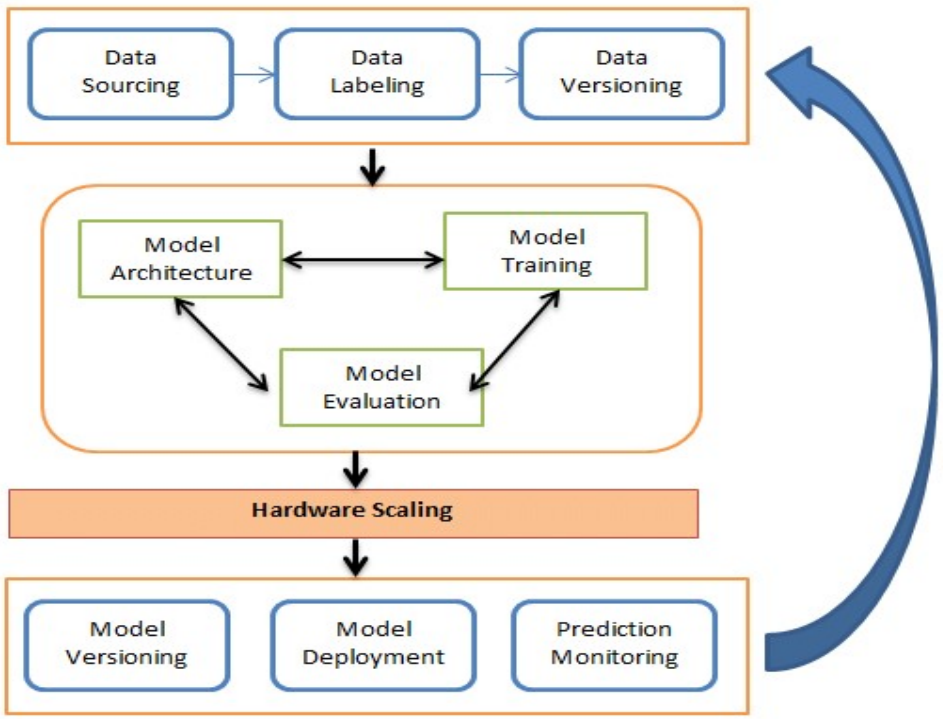


Fig. 5. Lifecycle of CNN Model

Data Sourcing: The data sourcing is considered as the essential step for any deep learning application. Sometimes historical data is readily available. Many times data can be available from the open-source datasets, raw data buying, search on internet for data or use simulated datasets.

Data Labeling: Supervised deep learning application which deals with image, text, video or audio required to annotate with application specific labels. This process is blend with the model training and deployment to maximize the advantage of trained deep learning model.

Data Versioning: Maintain the proper data version as more data evolves over time is an important step of the life cycle. For an instance, apply smart labelling process and retrain model again as data grows.

Scaling Hardware: Hardware plays an important role in model training as well as deployment. Moving from local development to large-scale experiment for model training, hardware scaling becomes essential.

Model Architecture: To train a model, it is required to select the appropriate model architecture or neural network. For standard classification problems one can use readily available state-of-the-art model or one can create customized model according to the need of the problem domain.

Model Training: To train the model, data with label required to feed into the model and to minimize the loss on the training update the model weights. Model can be train with set of hypermeters like learning rate, pre-processing steps and this process called hyper parameter tuning.

Model Evaluation: For deep learning model evaluation one metric for optimization is required. For given matric try to find model which performs best to generalize training data to the validation data.

Model Versioning: Model versioning is the intermediate step between model evaluation and model deployment. It is helpful in case when you required rolling back to the model worked well, when you find newest model version not performing up to the mark.

Model Deployment: If model for particular application is performing as per the expectations it can be put into the production.

Monitoring Predictions: After deploying model it is essential to keep eye on the predications it makes on the real data and get alerted regarding performance issues.

3. SYSTEM DESIGN

CHAPTER 3
SYSTEM DESIGN

3.1 INTRODUCTION

System design is a solution that "how to" approach the creation of a new system. It provides the understanding and procedural details necessary for implementing the proposed system.

3.2 BLOCK DIAGRAM

The block diagram of Convolutional Neural Network project "X-rays Based Pneumonia Detection using IBM Watson Studio".

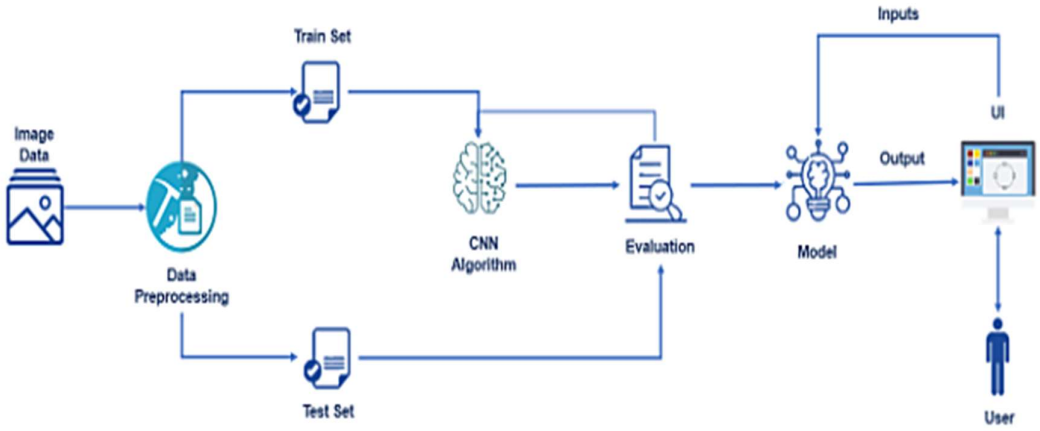


Fig. 6. Block diagram of CNN Model

Following are the step in this block diagram: -

- Download the dataset.
- Image Preprocessing
- Classify the dataset into train and test sets.
- Add the neural network layers.
- Load the trained images and fit the model.
- Test the model.
- Save the model and its dependencies.

- Build a Web application using flask that integrates with the model built.

3.2.1 Data Collection

Artificial Intelligence is a data hunger technology, it depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. In Convolutional Neural Networks, as it deals with images, we need training and testing data set. It is the actual data set used to train the model for performing various actions. The first activity is to download the dataset. The datasets can be collected from different open sources like kaggle.com, data.gov, UCI machine learning repository, etc.

In this project "X-rays Based Pneumonia Detection using IBM Watson Studio" the dataset contains 1812 images of chest X-rays (JPEG). It is divided into two folders, named train and test, that are used as training and testing data. The train folder for this experiment contains 1188 images and test folder 624 images with accuracy of 97%. Each of these two folders contains two subfolders containing images diagnosed as pneumonia or normal. The subfolder names represent the data labels.

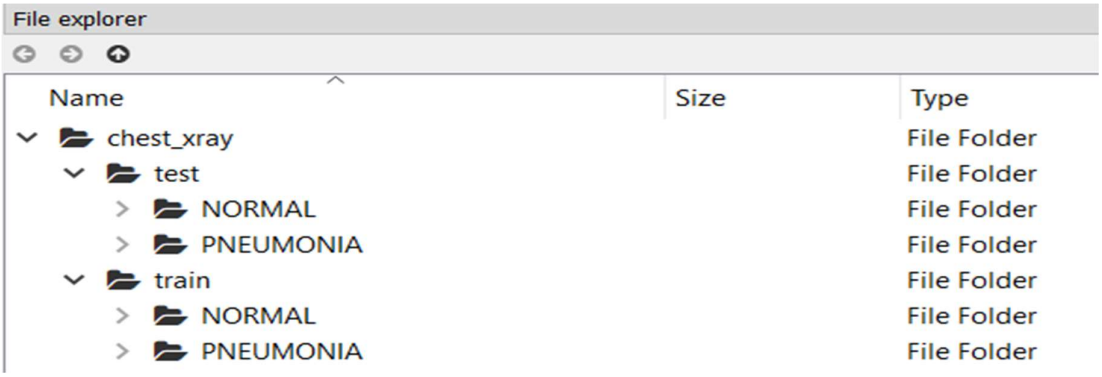
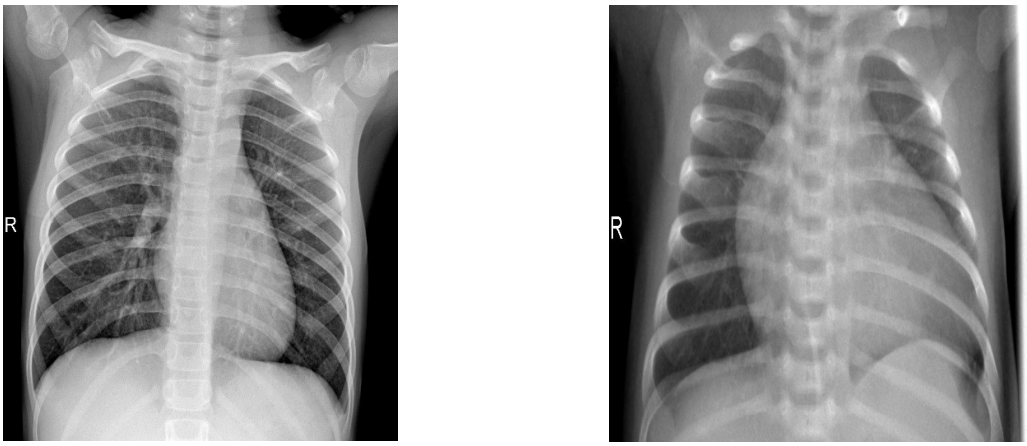


Fig. 7. Dataset folder structure



A. NORMAL B. PNEUMONIA

Fig 8. The dataset illustration: A) a chest X-ray labeled as pneumonia, and B) a chest X-ray labeled as normal

3.2.2 Image Preprocessing

Image Pre-processing includes the following main tasks

- Import ImageDataGenerator Library.
- Configure ImageDataGenerator Class.
- Applying ImageDataGenerator functionality to the trainset and test set.

Most of the time, the first step before building a model is the preprocessing of the imported data. The aims of pre-processing is to convert the input image data to into meaningful floating-point tensors for feeding into Convolutional Neural Networks. Just for the knowledge tensors are used to store data, they can be assumed as multidimensional arrays. A tensor representing a 224 X 224 image having 3 channels will have its dimensions (224, 224, 3). Currently, the data is stored on a drive as JPEG files, So the steps taken to achieve it are given below:

Algorithm:

- Read the picture files (stored in data folder).
- Decode the JPEG content to RGB grids of pixels with channels.
- Convert these into floating-point tensors for input to neural nets.
- Rescale the pixel values (between 0 and 255) to the [0, 1] interval (as training neural networks with this range gets efficient).

It may seem a bit fussy, but Keras has utilities to take over this whole algorithm and do the heavy lifting for you. Keras has a module with image-processing helping tools, located at keras.preprocessing.image. It contains the class ImageDataGenerator, which lets you quickly set up Python generators that can automatically turn image files on disk into batches of preprocessed tensors.

ImageDataGenerator class is instantiated and the configuration for the types of data augmentation

There are five main types of data augmentation techniques for image data; specifically:

- Image shifts via the width_shift_range and height_shift_range arguments.
- Image flips via the horizontal_flip and vertical_flip arguments.
- Image rotates via the rotation_range argument
- Image brightness via the brightness_range argument.
- Image zooms via the zoom_range argument.

In [1]:
```
# import the libraries as shown below

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
import numpy as np
from glob import glob
```

Image Data Agumentation¶

In [2]:
```
# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator
#performing data agumentation on train data
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)
#performing data agumentation on test data
test_datagen = ImageDataGenerator(rescale = 1./255)
```

Loading our data and performing data agumentation

In [3]:
```
# Make sure you provide the same target size as initialied for the image size
training_set = train_datagen.flow_from_directory(r'D:\AI-Pneumonia Detection\chest_xray\train',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 1188 images belonging to 2 classes.
```

In [4]:
```
test_set = test_datagen.flow_from_directory(r'D:\AI-Pneumonia Detection\chest_xray\test',
                                            target_size = (224, 224),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 624 images belonging to 2 classes.
```

3.2.3 Train and Test Set

Apply ImageDataGenerator functionality to train and test set. We can see that if figure training there are 1188 images belonging to 2 classes and for testing there are 624 images belonging to 2 classes.

Arguments:

- directory: Directory where the data is located. If labels are "inferred", it should contain subdirectories, each containing images for a class. Otherwise, the directory structure is ignored.
- batch_size: Size of the batches of data. Default: 32.
- target_size: Size to resize images to after they are read from disk.
- class_mode: 'categorical' means that the labels are encoded as a categorical vector (e.g. for categorical_crossentropy loss).

3.2.4 VGG16 CNN AND MODEL BUILDING

VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR(Imagenet) competition in 2014. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC(fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx) parameters.

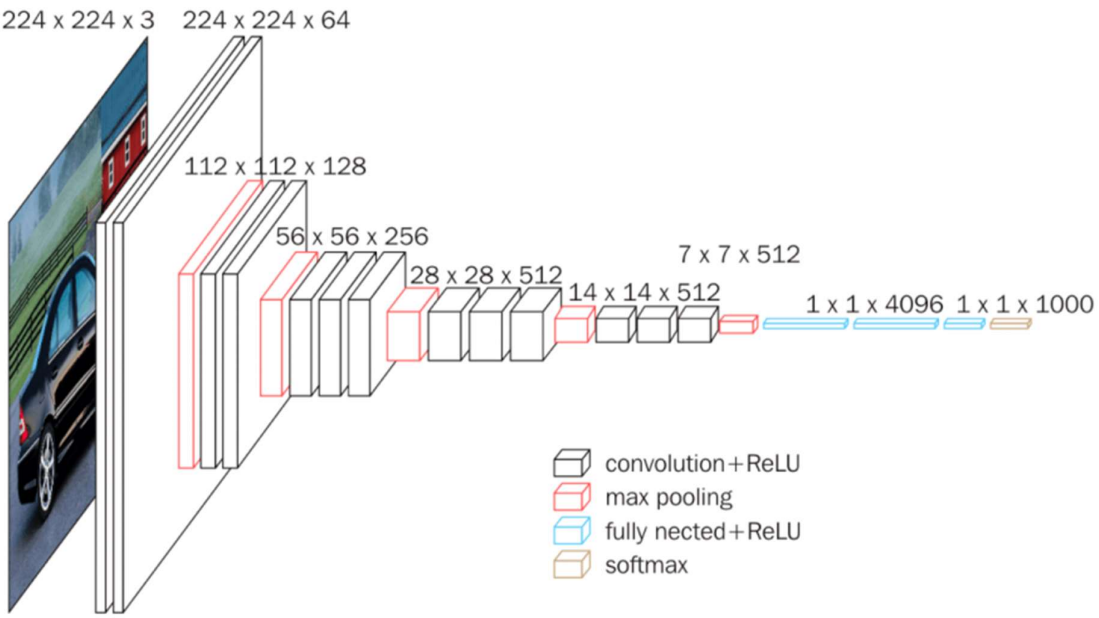


Fig. 9. VGG16 CNN layer process

Model Building

In [6]:
```
# re-size all the images to this
IMAGE_SIZE = [224, 224]

train_path = r'D:\AI-Pneumonia Detection\chest_xray\train'
valid_path = r'D:\AI-Pneumonia Detection\chest_xray\test'
```

In [7]:
```
# Import the Vgg 16 library as shown below and add preprocessing layer to the front of VGG
# Here we will be using imagenet weights

vgg16 = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

In [8]:
```
# don't train existing weights
for layer in vgg16.layers:
    layer.trainable = False
```

In [9]:
```
  # useful for getting number of output classes
folders = glob(r'D:\AI-Pneumonia Detection\chest_xray\train\*')
```

In [10]:
```
folders
```

Out[10]:
```
['D:\\AI-Pneumonia Detection\\chest_xray\\train\\NORMAL',
 'D:\\AI-Pneumonia Detection\\chest_xray\\train\\PNEUMONIA']
```

In [11]:
```
# our layers - you can add more if you want
x = Flatten()(vgg16.output)
```

In [12]:
```
prediction = Dense(len(folders), activation='softmax')(x)

# create a model object
model = Model(inputs=vgg16.input, outputs=prediction)
```

```
# create a model object
model = Model(inputs=vgg16.input, outputs=prediction)
```

In [13]:
```
# view the structure of the model
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 2)                 50178
=================================================================
Total params: 14,764,866
Trainable params: 50,178
Non-trainable params: 14,714,688
```

After adding all the required layers, the model is to be compiled. For this step, loss function, optimizer and metrics for evaluation can be passed as arguments.

- The compilation is the final step in creating a model. Once the compilation is done, we can move on to training phase. Loss function is used to find error or deviation in the learning process. Keras requires loss function during model compilation process.
- Optimization is an important process which optimize the input weights by comparing the prediction and the loss function. Here we are using adam optimizer

- Metrics is used to evaluate the performance of your model. It is similar to loss function, but not used in training process

Fit the model to the data using the generator, it is done using the fit_generator method, the Keras model needs to know how many samples to draw from the generator before declaring an epoch over. This is the role of the *steps_per_epoch* argument.
Now deciding the *steps_per_epoch* parameter, as we have total of 1188 training images and each batch is of size 32, hence, the steps_per_epoch will be 1188/ 32 = 37.12 = 38 (approximate)

Compiling the model

In [14]:
```
# tell the model what cost and optimization method to use
model.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)
```

Fit the model

In [17]:
```
# fit the model
# Run the cell. It will take some time to execute
r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=10,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

```
C:\Users\user\Anaconda3\lib\site-packages\keras\engine\training.py:1972: UserWarning: `Model.fit_generator` is deprecated and wi
ll be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/10
38/38 [==============================] - 699s 18s/step - loss: 0.4447 - accuracy: 0.8375 - val_loss: 0.3524 - val_accuracy: 0.86
38
Epoch 2/10
38/38 [==============================] - 681s 18s/step - loss: 0.1380 - accuracy: 0.9495 - val_loss: 0.2643 - val_accuracy: 0.89
74
Epoch 3/10
38/38 [==============================] - 684s 18s/step - loss: 0.1174 - accuracy: 0.9596 - val_loss: 0.3215 - val_accuracy: 0.87
50
Epoch 4/10
38/38 [==============================] - 680s 18s/step - loss: 0.1405 - accuracy: 0.9495 - val_loss: 0.2802 - val_accuracy: 0.89
58
Epoch 5/10
38/38 [==============================] - 482s 13s/step - loss: 0.1091 - accuracy: 0.9604 - val_loss: 0.2881 - val_accuracy: 0.89
42
Epoch 6/10
38/38 [==============================] - 425s 11s/step - loss: 0.0816 - accuracy: 0.9697 - val_loss: 0.3000 - val_accuracy: 0.88
62
Epoch 7/10
38/38 [==============================] - 423s 11s/step - loss: 0.0913 - accuracy: 0.9655 - val_loss: 0.2951 - val_accuracy: 0.88
94
Epoch 8/10
38/38 [==============================] - 429s 11s/step - loss: 0.0823 - accuracy: 0.9714 - val_loss: 0.3302 - val_accuracy: 0.87
98
Epoch 9/10
38/38 [==============================] - 450s 12s/step - loss: 0.0609 - accuracy: 0.9756 - val_loss: 0.3149 - val_accuracy: 0.87
66
Epoch 10/10
38/38 [==============================] - 431s 11s/step - loss: 0.0699 - accuracy: 0.9773 - val_loss: 0.3662 - val_accuracy: 0.87
34
```

After executing the epoch we get training accuracy of 97% . The weights are to be saved for future use. The weights are saved in as .**h5 file using save().**

Saving our model

```
In [27]: # save it as a h5 file
         from tensorflow.keras.models import load_model
         model.save('PneumoniaDetect.h5')
```

3.2.5 Application Building

After the model is built, we will be integrating it to a web application so that normal users can also use it. The users need to give the scan to know if pneumonia is present or not.

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the users where he/she has to uploads an image . The uploaded image is given to the saved model and prediction is showcased on the UI.

This section has the following tasks

- Building HTML Pages
- Building server-side script

Build Python Code

Build the flask file 'app.py' which is a web framework written in python for server-side scripting. Let's see step by step procedure for building the backend application.

- App starts running when the "__name__" constructor is called in main.
- render_template is used to return HTML file.
- "GET" method is used to take input from the user.
- "POST" method is used to display the output to the user

To run the code :

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type "python app.py" command
- Navigate to the localhost where you can view your web page

- Then it will run on localhost:5000
- Navigate to the localhost (http://127.0.0.1:5000/)where you can view your web page.

3.2.6 Flowchart of above block diagram

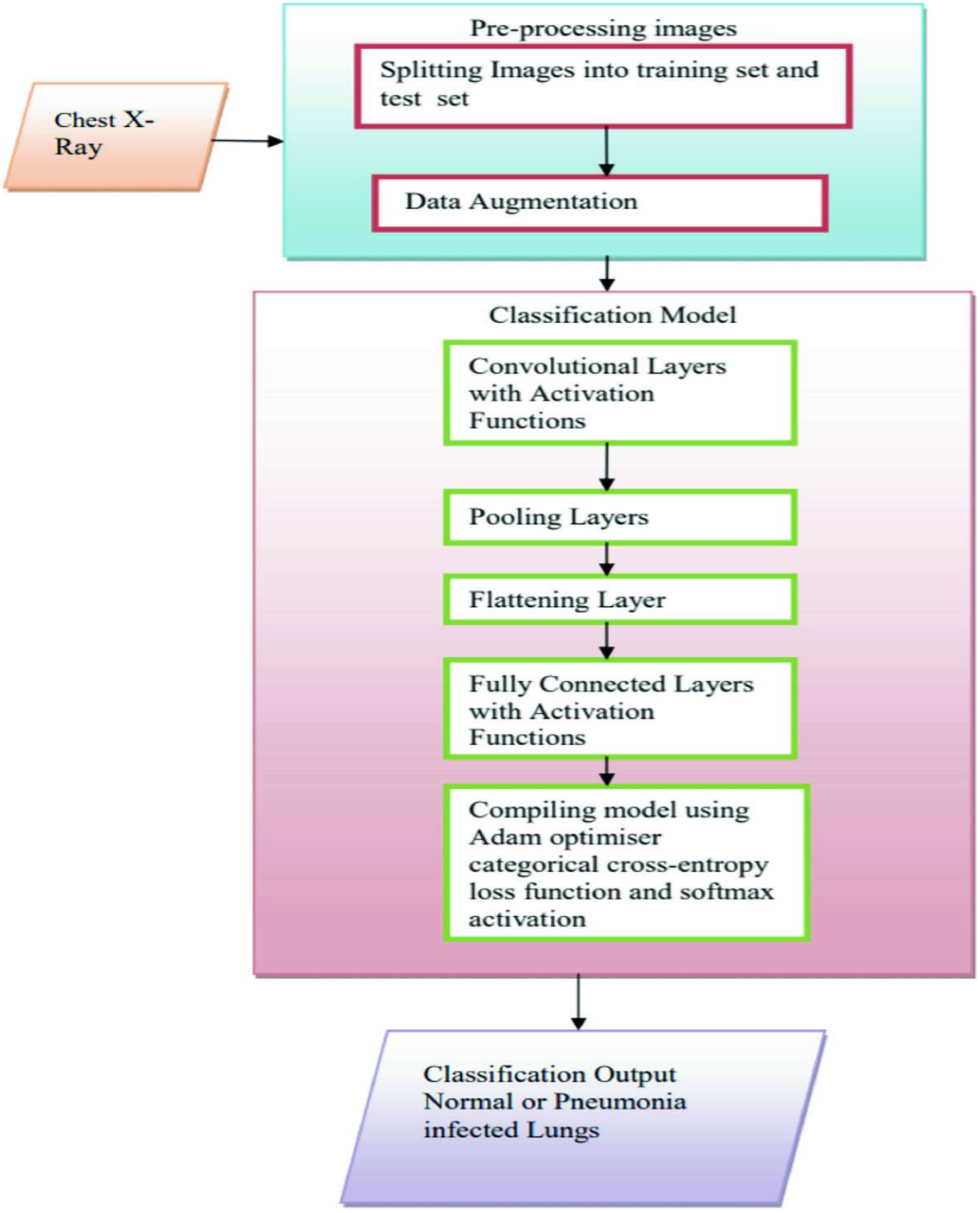


Fig.10 Flowchart of CNN process

3.2.7 Project structure

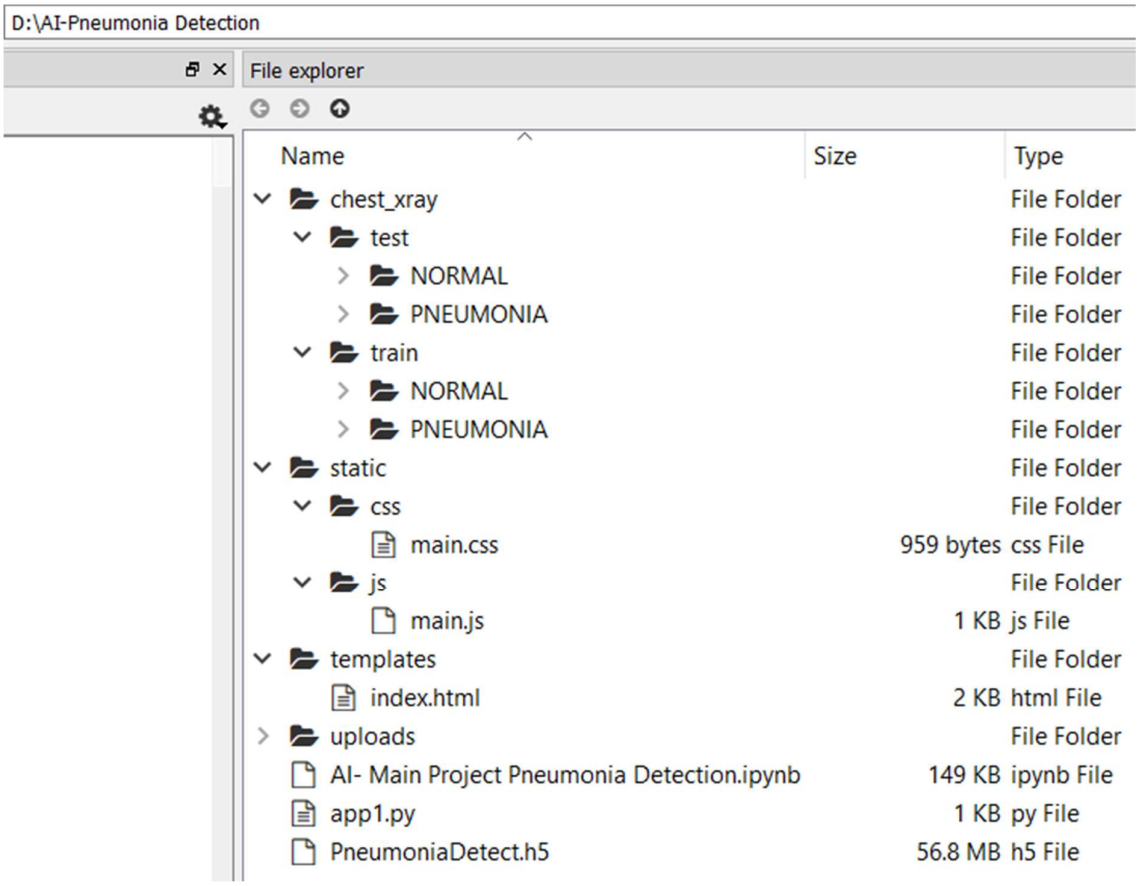


Fig.11 Project structure

3.3 INPUT DESIGN

Input design is the process of converting user-oriented input into a computer based format. The goal of the designing input is to make data entry as easy and free from error. In Android, input to the system is entered through activity. An activity is "any surface on which information is to be entered, the nature of which is determined by what is already on that surface." If the data going into the system is incorrect, then processing and output will magnify these errors. So designer should ensure that form is accessible and understandable by the user .End-users are people who communicate to the system frequently through the user interface, the design of the input screen should be according to their recommendations. The data is validated wherever it requires in the project. This ensures only correct data is entered to the system. GUI is the interface used in input design. All the input data are validated in the order and if any data violates any condition the use is warned by a message and asks to re-enter data. If the data

satisfies all the conditions then it is transferred to the appropriate tables in the database. This project uses text boxes and drop down to accept user input. If user enters wrong format then it shows a message to the user. User is never lift in confusion as to what is happening. Instead appropriate error messages and acknowledgments are displayed to the user.

3.4 OUTPUT DESIGN

Computer output is the most important one to the user. A major form of the output is the display of the information gathered by the system and the servicing the user requests to the system. Output generally refers to the results or information that is generated by the system. It can be in the form of operational documents and reports. Since some of the users of the system may not operate the system, but merely use the output from the system to aid them in decision-making, much importance is given to the output design. Output generation hence serves two main purposes, providing proper communication of information to the users and providing data in a form suited for permanent storage to be used later on. The output design phase consists of two stages, output definition and output specification. Output definition takes into account the types of outputs, its contents, formats, its frequency and its volume. The output specification describes each type of output in detail. The objective of the output design to convey the information of all the past activities, current status and emphasize important a quality output is one, which meets the requirements of the end user and presents the information.

4.SYSTEM ENVIRONMENT

CHAPTER-4
SYSTEM ENVIRONMENT

4.1 SOFTWARE ENVIRONMENT

Software environment is the term commonly used to refer to support an application. A software environment for a particular application could include the operating system, specific development tools or compiler.

4.2 SOFTWARE REQUIREMENT SPECIFICATION

Purpose and Scope

Purpose

To understand the nature of the program to be building the software engineers must understand the information domain for the software. Here the document specifies the software requirements of automating the functions. The document gives different software and hardware requirements of the system. This will help the users to understand there own needs. It will be the validation of all project.

Scope

This document is the only one that describes the requirements of the system to be developed. The software is helpful for both the rehabilitation institute and user.

4.2.1 TOOLS AND PLATFORMS

OVERVIEW OF WINDOWS 10

Windows 10 is a series of personal computer operating systems produced by Microsoft as part of its Windows NT family of operating systems. It is the successor to Windows 8.1, and was released to manufacturing on July 15, 2015, and broadly released for retail sale on July 29, 2015. Windows 10 receives new builds on an ongoing basis, which are available at no additional cost to users, in addition to additional test builds of Windows 10 which are available to Windows Insiders. The latest stable build of Windows 10 is Version 1903 (May 2019 Update). Devices in enterprise environments can receive these updates at a slower pace, or use

long-term support milestones that only receive critical updates, such as security patches, over their ten-year lifespan of extended support.

PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It was created by Guido van Rossum, and first released on February 20, 1991. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

IBM WATSON STUDIO

Professionals are putting AI to work to turn our most valuable resource — data — into new ways of doing business. With AI, we are no longer wrestling with data, but using it to recommend with confidence, accelerate research and discovery, and enrich interactions with customers on their terms.

Watson Studio accelerates the machine and deep learning workflows required to infuse AI into your business to drive innovation. It provides a suite of tools for data scientists, application developers and subject matter experts, allowing them to collaboratively connect to data, wrangle that data and use it to build, train and deploy models at scale. Successful AI projects require a

combination of algorithms + data + team, and a very powerful compute infrastructure. Until today, there was a gap between data experts and domain experts. Only highly technical professionals in IT could organize and make sense of the vast amounts of data. Only domain experts could successfully convert data into the rich knowledge needed by AI. But domain experts and IT professionals worked in silos, with different tools and no visibility to each others work. The result was AI that fell short in its promise to augment people's expertise.

Watson Studio closes the gap with a unified experience to create new insights from knowledge contained in the data. Watson Studio enables multidisciplinary teams across the organization to collaborate. We are convinced, after working with clients around the world, that rich collaboration is key unlocking the full potential of AI.

HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and last[3] major HTML version that is a World Wide Web Consortium (W3C) recommendation. The current specification is known as the HTML Living Standard. It is maintained by the Web Hypertext Application Technology Working Group (WHATWG), a consortium of the major browser vendors (Apple, Google, Mozilla, and Microsoft).

HTML5 was first released in a public-facing form on 22 January 2008,[2] with a major update and "W3C Recommendation" status in October 2014.[4][5] Its goals were to improve the language with support for the latest multimedia and other new features; to keep the language both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc., without XHTML's rigidity; and to remain backward-compatible with older software. HTML5 is intended to subsume not only HTML 4 but also XHTML 1 and DOM Level 2 HTML.[6]

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves, and rationalizes the markup available for documents and introduces markup and application programming interfaces (APIs) for complex web applications.[7] For the same reasons, HTML5 is also a candidate for cross-platform mobile applications because it includes features designed with low-powered devices in mind.

FLASK (WEB FRAMEWORK)

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Applications that use the Flask framework include Pinterest and LinkedIn.

Flask was created by Armin Ronacher of Pocoo, an international group of Python enthusiasts formed in 2004. When Ronacher and Georg Brandl created a bulletin board system written in Python in 2004, the Pocoo projects Werkzeug and Jinja were developed. In April 2016, the Pocoo team was disbanded and development of Flask and related libraries passed to the newly formed Pallets project.

Flask has become popular among Python enthusiasts. As of October 2020, it has second most stars on GitHub among Python web-development frameworks, only slightly behind Django, and was voted the most popular web framework in the Python Developers Survey 2018

In order to develop this project we need to install the following softwares/packages:

Step 1:

ANACONDA NAVIGATOR :

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code.

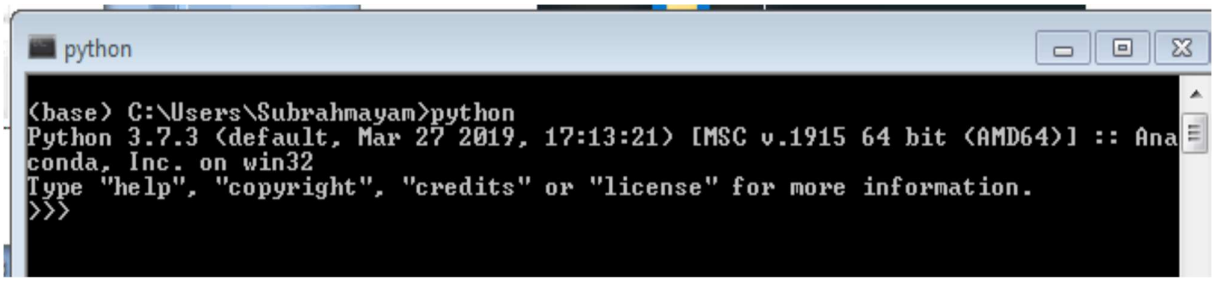
Install the anaconda prompt and open it. Anaconda is the Python distribution and the Anaconda Prompt is a command line shell.

On anaconda prompt, type python and hit [Enter]. The python command starts the Python interpreter, also called the Python REPL (for Read Evaluate Print Loop).

> >python

Note the Python version. You should see something like Python 3.7.3. With the interpreter running.

To close the Python interpreter, type exit() at the prompt >>>.



When you want to use the Python interpreter again, just click the Windows Start button and select the Anaconda Prompt and type python.

For this project, we will be using **Jupyter Notebbok** and **Spyder**

JUPYTER NOTEBOOK

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter.

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

SPYDER

Spyder, the Scientific Python Development Environment, is a free integrated development environment (IDE) that is included with Anaconda. It includes editing, interactive testing, debugging, and introspection features. Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community.Spyder is extensible with first-party and third-

party plugins,[6] includes support for interactive tools for data inspection and embeds Python-specific code

After you have installed Anaconda, start Spyder on Windows, macOS, or Linux by running the command spyder.
Spyder is also pre-installed in Anaconda Navigator, which is included in Anaconda.

KERAS

Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation.
Keras is relatively easy to learn and work with because it provides a python frontend with a high level of abstraction while having the option of multiple back-ends for computation purposes. This makes Keras slower than other deep learning frameworks, but extremely beginner-friendly.

Keras allows you to switch between different back ends. The frameworks supported by Keras are:

- Tensorflow
- Theano
- PlaidML
- MXNet
- CNTK (Microsoft Cognitive Toolkit)

Out of these five frameworks, TensorFlow has adopted Keras as its official high-level API. Keras is embedded in TensorFlow and can be used to perform deep learning fast as it provides inbuilt modules for all neural network computations. At the same time, computation involving tensors, computation graphs, sessions, etc can be custom made using the Tensorflow Core API, which gives you total flexibility and control over your application and lets you implement your ideas in a relatively short time.

Why Do We Need Keras?

- Keras is an API that was made to be easy to learn for people. Keras was made to be simple. It offers consistent & simple APIs, reduces the actions required to implement common code, and explains user error clearly.
- Prototyping time in Keras is less. This means that your ideas can be implemented and deployed in a shorter time. Keras also provides a variety of deployment options depending on user needs.
- Languages with a high level of abstraction and inbuilt features are slow and building custom features in then can be hard. But Keras runs on top of TensorFlow and is relatively fast. Keras is also deeply integrated with TensorFlow, so you can create customized workflows with ease.
- The research community for Keras is vast and highly developed. The documentation and help available are far more extensive than other deep learning frameworks.
- Keras is used commercially by many companies like Netflix, Uber, Square, Yelp, etc which have deployed products in the public domain which are built using Keras.

Apart from this, Keras has features such as :

- It runs smoothly on both CPU and GPU.
- It supports almost all neural network models.
- It is modular in nature, which makes it expressive, flexible, and apt for innovative research.

TENSORFLOW

TensorFlow is a Python library for fast numerical computing created and released by Google.

It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.
TensorFlow is an open source library for fast numerical computing.

It was created and is maintained by Google and released under the Apache 2.0 open source license. The API is nominally for the Python programming language, although there is access to the underlying C++ API.

Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project.
It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines.

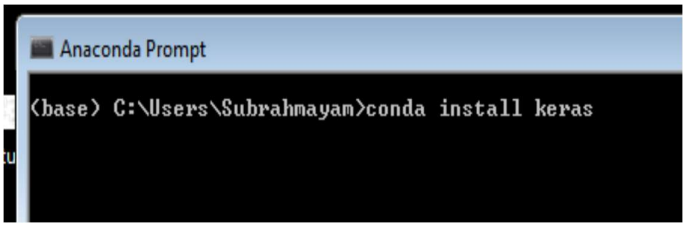
OPENCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency.

To install Keras, TensorFlow and OpenCV

Open anaconda prompt and execute **conda install keras** to install keras library and press enter.



Open anaconda prompt and execute **conda install tensorflow** to install tensorflow library and press enter

Open anaconda prompt and execute **conda install opencv-python** to install Open cv library and press enter.

4.3 AI Hardware Requirements:

- **Operating system:** window 7 and above with 64bit
- **Processor Type** -Intel Core i3-3220
- **RAM:** 4Gb and above
- **Hard disk:** min 100GB

5. SYSTEM IMPLEMENTATION AND TESTING

CHAPTER-5

SYSTEM IMPLEMENTATION AND TESTING

5.1 CODING

CODING STANDARDS

Coding standards are important because they lead to greater consistency within code of all developers. Consistency leads to code that is easier to understand, which in turn results in turn result in a code, which is easier to develop and maintain. Code that difficult to understand and maintain runs the risks of being scrapped rewritten.

5.2 TESTING AND VERIFICATION PROCEDURES

Unit Testing

Unit testing is a concept that would be familiar to people coming from software development. It is a very useful technique that can help you prevent obvious errors and bugs in your code. It involves testing individual units of the source code, such as functions, methods, and class to ascertain that they meet the requirements and have expected behaviour. Unit tests are usually small and don't take much time to execute. The tests have a wide range of inputs often including boundary and edge cases. The outputs of these inputs are usually calculated by the developer manually to test the output of the unit being tested. For example for an adder function, we would have test cases something like the following.

You test cases with positive inputs, inputs with zero, negative inputs, positive and negative inputs. If the output of our function/method being tested would be equal to the outputs defined in the unit test for all the input cases, your unit would pass the test otherwise it would fail. You would know exactly which test case failed. Which can be further investigated to find out the problem. This is an awesome sanity check to have in your code. Especially if multiple developers are working on a large project. Imagine someone wrote a piece of code based on certain assumptions and data sizes and a new developer changes something in the codebase

which no longer meets those assumptions. Then the code is bound to fail. Unit tests allow avoiding such situations.

Following are some of the benefits of unit testing.

Forces you to write modular and reusable code with clearly defined inputs and outputs. As a result, your code would be easier to integrate.
Increased confidence in changing/maintaining code. It helps to identify bugs introduced by a code change.
Improved confidence in the unit itself since if it passes the unit tests we are sure that there is nothing obviously wrong with the logic and the unit is performing as intended.
Debugging becomes easier since you would know which unit failed as well as the particular test cases which failed.

Integration Testing

Data can be lost across an interface, one module can have an adverse effect on the other sub-functions, when combined they may not perform the desired functions. Integrated testing is the systematic testing to uncover the errors within the interface. This testing is done with simple data and the developed system has run successfully with this simple data. The need for integrated system is to the overall system performance.

The Modules of this project are connected and tested. After splitting the programs into units, the units were tested together to see the defects between each module and function. It is testing to one or more modules or functions together with the intent of interface defects between the modules or functions. Testing completed as part of unit or functional testing, integration testing can involve putting together of groups of modules and functions with the goal of completing and verifying meets the system requirements.

System Testing

System testing focuses on testing the system as a whole. System Testing is a crucial step in Quality Management Process. In the Software Development Life Cycle, System Testing is the first level where the System is tested as a whole. The System is tested to verify whether it meets the functional and technical requirements.

User Acceptance Testing

The system was tested by a small client community to see if the program met the requirements the analysis stage. It was found to be satisfactory. In this phase, the system is fully tested by

the client community against the requirements in the analysis and design stages, corrections are made as required, and the production system is built. User acceptance of the system is key factor for success of the system.

Types of acceptance test

The software application may use different users on different way & it impossible to developer or tester to predict what all possible scenarios or test data end user will use & how customer actually use the software application. So most of software venders are use the term like Alpha testing and Beta Testing which help to uncover the errors that may occurs in the actual test environment. In this testing method the software application release over limited end users rather than testing professionals to get feedback from them.

Alpha Testing

Alpha testing is conducted by Customer at the developer‟s site, it is performed by potential users like developer, end users or organization users before it is released to external customers & report the defects found while Alpha testing. This software product testing is not final version of software application, after fixing all reported bug (after bug triage) the new version of software application will release.

Sometimes the Alpha Testing is carried out by client or an outsider with the attendance of developer and tester. The version of the release on which Alpha testing is perform is called “Alpha Release”.

Beta Testing

Most if times we have the sense of hearing term “Beta release/version”, so it is linked to Beta Testing. Basically the beta testing is to be carried out without any help of developers at the end user‟s site by the end users &, so it is performed under uncontrolled environment. Beta testing is also known as Field testing. This is used to get feedback from the market. This testing is conducted by limited users & all issues found

during this testing are reported on continuous basis which helps to improve the system. Developers are taking actions on all issues reported in beta testing after bug triage & then the software application is ready for the final release. The version release after beta testing is called “Beta Release“.

SOFTWARE TESTING

Software testing is critical element of software quality assurance and represents ultimate review of specification design and coding system with testing is actually a series of different task whose primary objective is to fully exercise computer-based systems through successfully, it

will uncover error in software. Testing is a process of executing a program with intension of finding an error, Good test case is one that has a high probability of finding undiscovered error.

VALIDATION TESTING

In validation testing, all the relevant fields are checked to whether they contain data and also checks whether they hold the right data format guarantee that all the independent path with in a module have been exercised at least once

o Exercise all the logical decisions on their true or false sides.

o Exercise all loops at their boundaries and within their operational bounds.

TEST CASES

A specific set of steps and data along with expected results for a particular test objective. A test case should only test one limited subset of a future or functionality. Test case documents for each functionality/testing area of our project is written, reviewed and maintained separately. Test cases that check error conditions are written separately from the functional test cases and should have steps to verify the error messages.

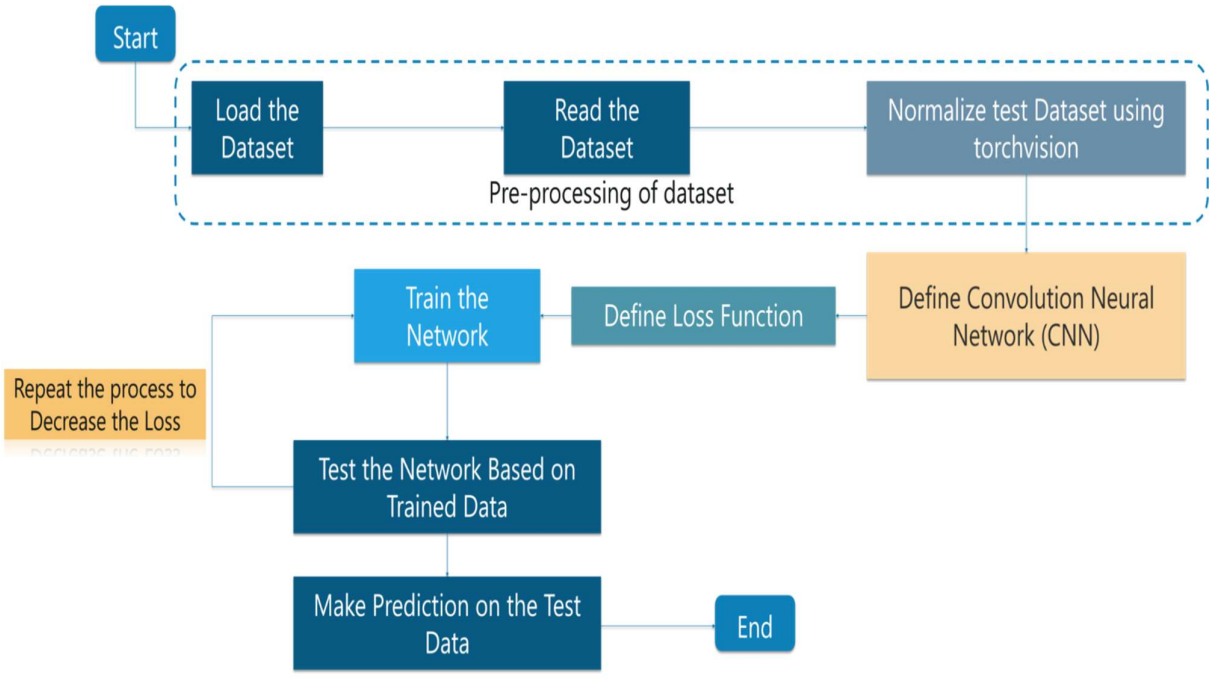


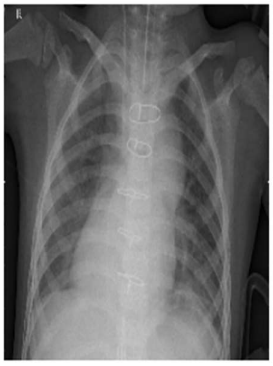
Fig.12 Steps of Dataset train, test and prediction

Predicitng our results

In [28]:
```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("PneumoniaDetect.h5") #loading the model for testing
```

In [29]:
```
img=image.load_img(r'D:\AI-Pneumonia Detection\chest_xray\train\PNEUMONIA\person2_bacteria_3.jpeg',target_size=(224,224))
img
```

Out[29]:



In [30]:
```
x=image.img_to_array(img)
print(x.shape)
```

```
(224, 224, 3)
```

In [31]:
```
x=np.expand_dims(x,axis=0)
print(x.shape)
```

```
(1, 224, 224, 3)
```

In [32]:
```
predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis=1)
classes_x
```

Out[32]:
```
array([1], dtype=int64)
```

In [33]:
```
index=["Normal","Pneumonic"]
```

In [34]:
```
result=str(index[classes_x[0]])
result
```

Out[34]:
```
'Pneumonic'
```

In [35]:
```
img=image.load_img(r'D:\AI-Pneumonia Detection\chest_xray\train\NORMAL\IM-0149-0001.jpeg',target_size=(224,224))
img
```

Out[35]:

In [36]:
```
x=image.img_to_array(img)
print(x.shape)
```

```
(224, 224, 3)
```

In [37]:
```
x=np.expand_dims(x,axis=0)
print(x.shape)
```

```
(1, 224, 224, 3)
```

In [38]:
```
predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis=1)
classes_x
```

Out[38]:
```
array([0], dtype=int64)
```

In [39]:
```
index=["Normal","Pneumonic"]
```

In [40]:
```
result=str(index[classes_x[0]])
result
```

Out[40]:
```
'Normal'
```

5.3 SYSTEM IMPLEMENTATIONS

Implementation is the process of personnel check out, install the required equipment and application and train user accordingly. Depending on the size of the organization and its requirements the implementation can be divided into three:

Stage Implementation

Here system is implemented in stages. The whole system is not implemented at once. Once the user starts working with system and is familiar with it, then a stage is introduced and implemented. Also the system is usually updated, regularly until a final system is sealed.

Direct Implementation

The proposed new system is implemented directly and the user starts working on the new System. The shortcoming, if any, faced are then rectified later.

Parallel Implementation

The old and the new system are not used simultaneously. This helps in comparison of the results from the two systems. Once the user is satisfied and his intended objectives are achieved by the new system, he stops using the old one.

In my project I have used direct implementation method. The client (doctors) is given with fully developed system. System developed by using deep learning convolutional neural network to process chest X-ray images in order to support the decision-making process in determining the correct diagnosis. This model has the task to help with a classification problem that is detecting whether a chest X-ray shows changes consistent with pneumonia or not, and classifying the X-ray images in two groups depending on the detection results. It help the doctors to predict the pneumonia disease more accurately using a deep learning model. The objective is not only to help the doctors but also to the patients to precisely predict pneumonia.

6. SYSTEM MAINTENANCE

CHAPTER-6

SYSTEM MAINTENANCE

6.1 MAINTENANCE

Software Maintenance is the process of modifying a software product after it has been delivered to the client. The main purpose of software maintenance is to modify and update software application after delivery to correct faults and to improve performance.

Need for Maintenance –Software Maintenance must be performed in order to:

- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.
- Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.
- Migrate legacy software.
- Retire software.

Categories of Software Maintenance –Maintenance can be divided into the following:

- **Corrective maintenance** Corrective maintenance of a software product may be essential either to rectify some bugs observed while the system is in use, or to enhance the performance of the system.
- **Adaptive maintenance:** This includes modifications and updating when the customers need the product to run on new platforms, on new operating systems, or when they need the product to interface with new hardware and software.
- **Perfective maintenance:** A software product needs maintenance to support the new features that the users want or to change different types of functionalities of the system according to the customer demands.

 Preventive maintenance: This type of maintenance includes modifications and updations to prevent future problems of the software. It goals to attend problems, which are not significant at this moment but may cause serious issues in future.

Reverse Engineering – Reverse Engineering is processes of extracting knowledge or design information from anything man-made and reproducing it based on extracted information. It is also called back Engineering.

Software Reverse Engineering – Software Reverse Engineering is the process of recovering the design and requirements specification of a product from an analysis of its code. It is becoming important, since several existing software products, lack proper documentation, are highly unstructured, or their structure has degraded through a series of maintenance efforts.

7.SYSTEM SECURITY MEASURES

CHAPTER-7
SYSTEM SECURITY MEASURES

7.1 INTRODUCTION

Security applied to computing devices such as computers and smartphones, as well as computer networks such as private and public networks, including the whole Internet. The field covers all the processes and mechanisms by which digital equipment, information and services are protected from unintended or unauthorized access, change or destruction, and is of growing importance in line with the increasing reliance on computer systems of most societies worldwide. Computer security includes measures taken to ensure the integrity of files stored on a computer or server as well as measures taken to prevent unauthorized access to stored data, by securing the physical perimeter of the computer equipment, authentication of users or computer accounts accessing the data, and providing a transmission.

The variety of threats combined with the rapid development of new threats has made cyber insecurity and the removal of information assurance the 'status quo'. As long as man continues to use the computer, man will also takes interest in manipulating, modifying, creating and bypassing 'rules' and 'security standards.'

7.2 OPERATING SYSTEM LEVEL SECURITY

Operating system security (OS security) is the process of ensuring OS integrity, confidentiality and availability. OS security refers to specified steps or measures used to protect the OS from threats, viruses, worms, malware or remote hacker intrusions. OS security encompasses all preventive-control techniques, which safeguard any computer assets capable of being stolen, edited or deleted if OS security is compromised. OS security encompasses many different techniques and methods which ensure safety from threats and attacks. OS security allows different applications and programs to perform required tasks and stop unauthorized interference. OS security may be approached in many ways, including adherence to the following:

- Performing regular OS patch updates
- Installing updated antivirus engines and software
- Scrutinizing all incoming and outgoing network traffic through a firewall

 Creating secure accounts with required privileges only (i.e., user management)

7.3 SYSTEM LEVEL SECURITY

System-level security refers to the architecture, policy and processes that ensure data and system security on individual computer systems. It facilitates the security of standalone and/or network computer systems/servers from events and processes that can exploit or violate its security or stature.

System-level security is part of a multi-layered security approach in which information security (IS) is implemented on an IT infrastructure's different components, layers or levels. System-level security is typically implemented on end-user computer and server nodes. It ensures that system access is granted only to legitimate and trusted individuals and applications. The key objective behind system-level security is to keep system secure, regardless of security policies and processes at other levels. If other layers or levels are breached, the system must have the ability to protect itself.

Methods used to implement system-level security are user/ID login credentials, antivirus and system-level firewall applications.

8.FUTURE SCOPE AND FURTHER ENHANCEMENT

CHAPTER-8

FUTURE SCOPE AND FURTHER ENHANCEMENT

INTRODUCTION

In the future, it is intended that larger datasets will also be trained using the models presented in the project. It is also expected that neural network models based on GAN (Generative Adversarial Networks), would also be trained and compared with the existing models. Further research steps will include experimenting with various preprocessing and CNN configurations, data augmentation techniques, as well as using additional X-ray datasets with additional data labels showing other pathologies.

9. CONCLUSION

CHAPTER-9
CONCLUSION

CONCLUSION

Thus, it is concluded that in this project "X-rays Based Pneumonia Detection using IBM Watson Studio" the CNN model can be effectively used by medical officers for diagnostic purposes for early detection of pneumonia in children as well as adults. A large number of X-ray images can be processed very quickly to provide highly precise diagnostic results, thus helping healthcare systems provide efficient patient care services and reduce mortality rates. A web application is integrated into the model, from where you can upload an x-ray image and see the analysed results on User Interface. A GUI based application which uses a custom CNN model to predict if uploaded chest x-ray image is normal or pneumonic. System will read the image uploaded by the user, augment it and will use the saved custom model to detect whether the disease is present or not in the patient and thus display the result in a user-friendly language.

Pneumonia can be serious so it's important to get treatment quickly. The main treatment for pneumonia is antibiotics, along with rest and drinking plenty of water. Treatment depends on how severe your pneumonia is. Treatment with antibiotics should be started as soon as possible after diagnosis. Therefore, a timely diagnosis of pneumonia is very important. So, more automations must be brought in this field, and make the detection of diseases more effective and accurate.

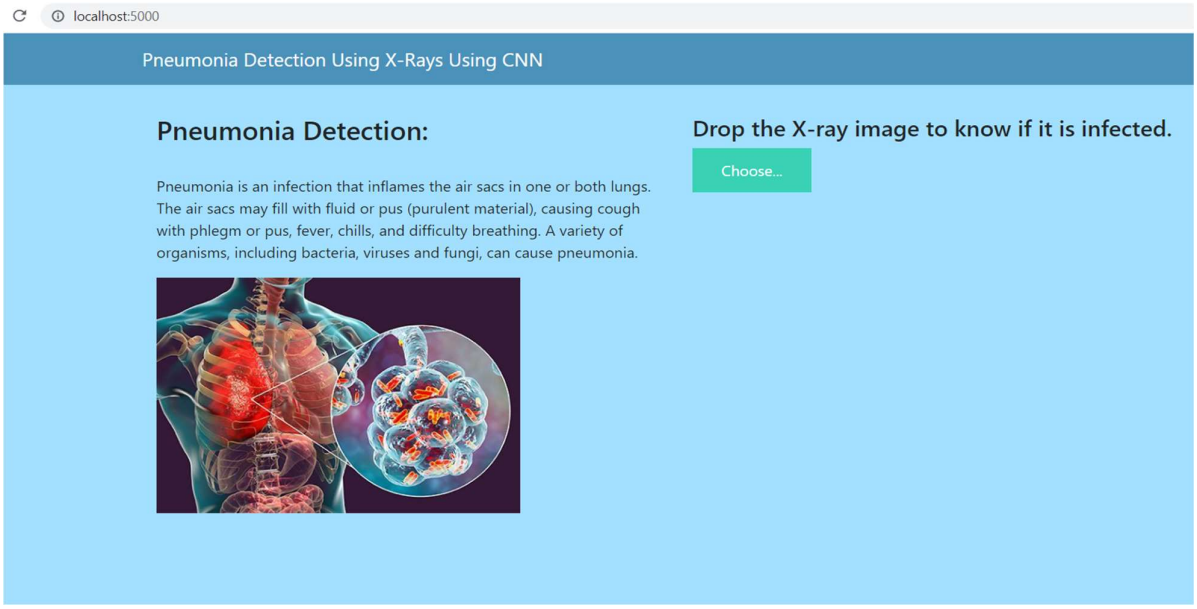
10. BIBLIOGRAPHY

CHAPTER-10
BIBLIOGRAPHY

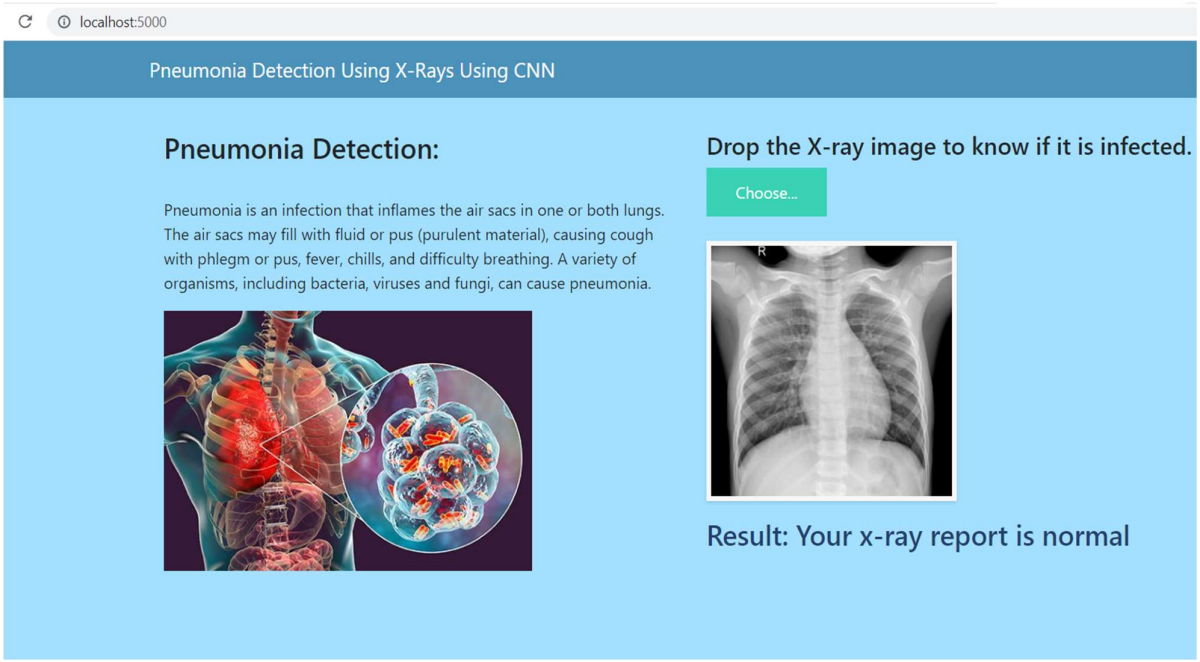
- SmartInternz.com
- Stackoverflow.com
- https://www.kaggle.com/madz2000/pneumonia-detection-using-cnn
- Jaiswal, A.K., Tiwari, P., Kumar, S., Gupta, D., Khanna, A., Rodrigues, J.J.: Identifying pneumonia in chest x-rays: a deep learning approach. Measurement **145**, 511–518 (2019)
- Kim, D.H., MacKinnon, T.: Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks. Clin. Radiol. **73**(5), 439–445 (2018)
- Bernal, J., Kushibar, K., Asfaw, D.S., Valverde, S., Oliver, A., Martí, R., Lladó, X.: Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. Artif. Intell. Med. **95**, 64–81 (2019)
- Arthur, F., Hossein, K.R.: Deep learning in medical image analysis: a third eye for doctors. J. Stomatology Oral Maxillofac. Surg.
- Rubin, J., Sanghavi, D., Zhao, C., Lee, K., Qadir, A., Xu-Wilson, M.: Large Scale Automated Reading of Frontal and Lateral Chest X-Rays Using Dual Convolutional Neural Networks(2018).

11. APPENDIX

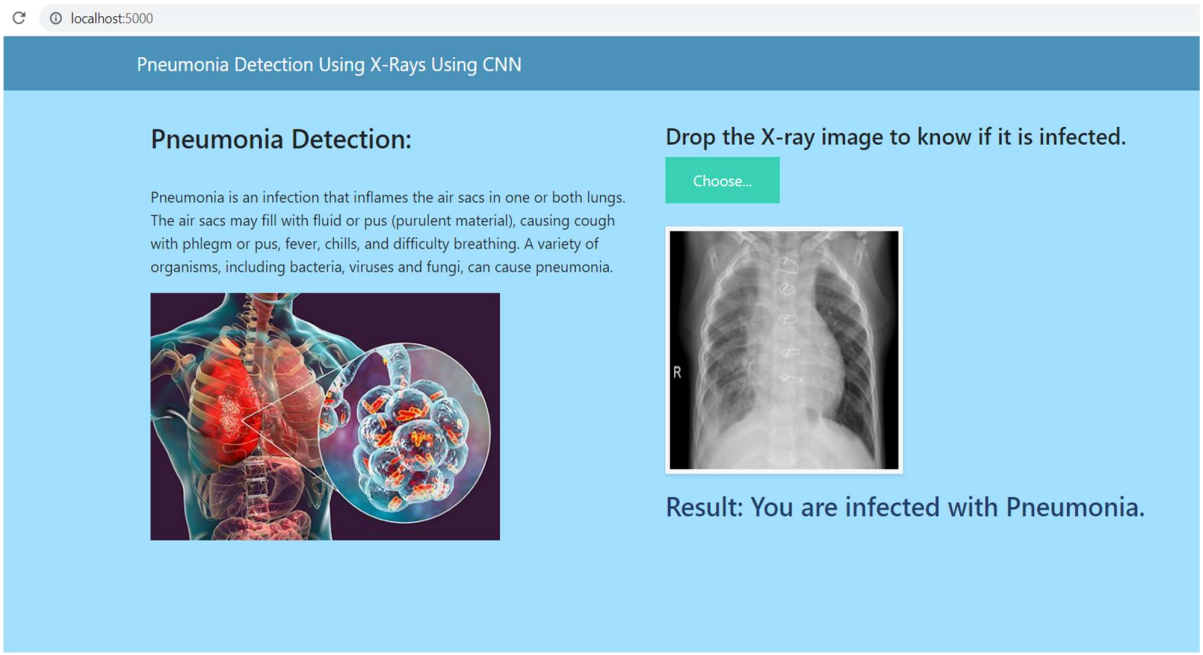
11.1 SCREEN SHOTS



Output -1



Output – 2



11.2 Sample Code

Jupyter Notebook

Detecting Building defects using VGG16

Importing necessary libraries

 # import the libraries as shown below

from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.preprocessing import image

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
import numpy as np
from glob import glob
```

Image Data Agumentation¶

```
# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator
#performing data agumentation on train data
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)
#performing data agumentation on test data
test_datagen = ImageDataGenerator(rescale = 1./255)
```

Loading our data and performing data agumentation

```
# Make sure you provide the same target size as initialied for the image size

training_set = train_datagen.flow_from_directory(r'D:\AI-Pneumonia Detection\chest_xray\tr
ain',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

Found 1188 images belonging to 2 classes.

```
test_set = test_datagen.flow_from_directory(r'D:\AI-Pneumonia Detection\chest_xray\test',
                                            target_size = (224, 224),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

Found 624 images belonging to 2 classes.

```
print(training_set.class_indices)#checking the number of classes
{'NORMAL': 0, 'PNEUMONIA': 1}
```

Model Building

```
# re-size all the images to this
IMAGE_SIZE = [224, 224]

train_path = r'D:\AI-Pneumonia Detection\chest_xray\train'
valid_path = r'D:\AI-Pneumonia Detection\chest_xray\test'

# Import the Vgg 16 library as shown below and add preprocessing layer to the front of VGG
# Here we will be using imagenet weights

vgg16 = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)



# don't train existing weights
for layer in vgg16.layers:
    layer.trainable = False


  # useful for getting number of output classes
folders = glob(r'D:\AI-Pneumonia Detection\chest_xray\train\*')

folders

['D:\\AI-Pneumonia Detection\\chest_xray\\train\\NORMAL',
 'D:\\AI-Pneumonia Detection\\chest_xray\\train\\PNEUMONIA']
```

```
# our layers - you can add more if you want
x = Flatten()(vgg16.output)


prediction = Dense(len(folders), activation='softmax')(x)

# create a model object
model = Model(inputs=vgg16.input, outputs=prediction)


# view the structure of the model
model.summary()
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
```

```
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 2)                 50178
=================================================================
Total params: 14,764,866
Trainable params: 50,178
Non-trainable params: 14,714,688
_________________________________________________________________
```

Compiling the model

```
# tell the model what cost and optimization method to use
model.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
```

```
  metrics=['accuracy']
)
```

Fit the model

```
# fit the model
# Run the cell. It will take some time to execute
r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=10,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
C:\Users\user\Anaconda3\lib\site-packages\keras\engine\training.py:1972: UserWarning: `M
odel.fit_generator` is deprecated and will be removed in a future version. Please use `Model.f
it`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '


Epoch 1/10
38/38 [==============================] - 699s 18s/step - loss: 0.4447 - accuracy: 0
.8375 - val_loss: 0.3524 - val_accuracy: 0.8638
Epoch 2/10
38/38 [==============================] - 681s 18s/step - loss: 0.1380 - accuracy: 0
.9495 - val_loss: 0.2643 - val_accuracy: 0.8974
Epoch 3/10
38/38 [==============================] - 684s 18s/step - loss: 0.1174 - accuracy: 0
.9596 - val_loss: 0.3215 - val_accuracy: 0.8750
Epoch 4/10
38/38 [==============================] - 680s 18s/step - loss: 0.1405 - accuracy: 0
.9495 - val_loss: 0.2802 - val_accuracy: 0.8958
Epoch 5/10
```

```
38/38 [==============================] - 482s 13s/step - loss: 0.1091 - accuracy: 0
.9604 - val_loss: 0.2881 - val_accuracy: 0.8942
Epoch 6/10
38/38 [==============================] - 425s 11s/step - loss: 0.0816 - accuracy: 0
.9697 - val_loss: 0.3000 - val_accuracy: 0.8862
Epoch 7/10
38/38 [==============================] - 423s 11s/step - loss: 0.0913 - accuracy: 0
.9655 - val_loss: 0.2951 - val_accuracy: 0.8894
Epoch 8/10
38/38 [==============================] - 429s 11s/step - loss: 0.0823 - accuracy: 0
.9714 - val_loss: 0.3302 - val_accuracy: 0.8798
Epoch 9/10
38/38 [==============================] - 450s 12s/step - loss: 0.0609 - accuracy: 0
.9756 - val_loss: 0.3149 - val_accuracy: 0.8766
Epoch 10/10
38/38 [==============================] - 431s 11s/step - loss: 0.0699 - accuracy: 0
.9773 - val_loss: 0.3662 - val_accuracy: 0.8734
```

Saving our model

```
# save it as a h5 file

from tensorflow.keras.models import load_model

model.save('PneumoniaDetect.h5')
```

Predicitng our results

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("PneumoniaDetect.h5") #loading the model for testing
```

```
img=image.load_img(r'D:\AI-Pneumonia Detection\chest_xray\train\PNEUMONIA\person2
_bacteria_3.jpeg',target_size=(224,224))
img
```

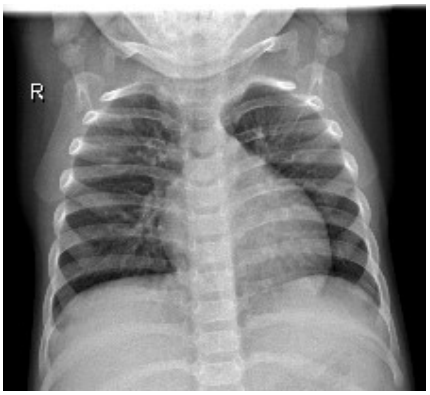
```
x=image.img_to_array(img)
print(x.shape)
(224, 224, 3)

x=np.expand_dims(x,axis=0)
print(x.shape)
(1, 224, 224, 3)

predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis=1)
classes_x
array([1], dtype=int64)

index=["Normal","Pneumonic"]
result=str(index[classes_x[0]])
result
'Pneumonic'
```

```
img=image.load_img(r'D:\AI-Pneumonia Detection\chest_xray\train\NORMAL\IM-0149-00
01.jpeg',target_size=(224,224))
img
```




```
x=image.img_to_array(img)
print(x.shape)
(224, 224, 3)

x=np.expand_dims(x,axis=0)
print(x.shape)
(1, 224, 224, 3)
predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis=1)
classes_x
array([0], dtype=int64)

index=["Normal","Pneumonic"]
result=str(index[classes_x[0]])
result
'Normal'
```

Spyder

```
import numpy as np
import os
from tensorflow.keras.models import load_model
```

```
from tensorflow.keras.preprocessing import image
from flask import Flask , request, render_template

app = Flask(__name__)

model = load_model("PneumoniaDetect.h5")

@app.route('/')
def index():
    return render_template(r'index.html')

@app.route('/predict',methods = ['GET','POST'])
def upload():
    if request.method == 'POST':
        f = request.files['image']
        print("current path")
        basepath = os.path.dirname(__file__)
        print("current path", basepath)
        filepath = os.path.join(basepath,'uploads',f.filename)
        print("upload folder is ", filepath)
        f.save(filepath)
        img = image.load_img(filepath,target_size=(224,224))
        x = image.img_to_array(img)
        print(x)
        a=np.expand_dims(x,axis=0)
        print(a)
        predict_x=model.predict(a)
        classes_x=np.argmax(predict_x,axis=1)
        print("prediction",classes_x)
        index = ['Your x-ray report is normal','You are infected with Pneumonia.']
        text = str(index[classes_x[0]])
    return text
if __name__ == '__main__':
    app.run()
```

index.html

```
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie=edge">
   <title>Pneumonia Detection Using X-Rays</title>
   <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">
   <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
   <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
   <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
   <link href="{{ url_for('static', filename='css/main.css') }}" rel="stylesheet">
        <style>

        .bg-dark {
                background-color:      #4B92BB!important;
        }
        #result {
                color: #0a1c4ed1;
        }
        body
{
   background-image:
url("
B6zFdcAAAAA1BMVEWj4P8POMKCAAAASElEQVR4nO3BMQEAAADCoPVPbQwfo
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAIC3AcUIAAFkqh/QAAAAAElFTkSuQmCC");
   background-size: cover;
}
</style>
</head>
```

```
<body>

  <nav class="navbar navbar-dark bg-dark">
    <div class="container">
      <a class="navbar-brand" href="#">Pneumonia Detection Using X-Rays Using
CNN</a>
    </div>
  </nav>
  <div class="container">
    <div id="content" style="margin-top:2em">
          <div class="container">
           <div class="row">
              <div class="col-sm-6 bd">
               <h3>Pneumonia Detection: </h3>
               <br>
               <p>Pneumonia is an infection that inflames the air sacs in one or both
lungs. The air sacs may fill with fluid or pus (purulent material), causing cough with phlegm
or pus, fever, chills, and difficulty breathing. A variety of organisms, including bacteria,
viruses and fungi, can cause pneumonia.</p>
                    <img
src="https://images.medicinenet.com/images/article/main_image/pneumonia-3.jpg"
height="50%",width="20%">
              </div>
              <div class="col-sm-6">
                    <div>
                         <h4>Drop the X-ray image to know if it is
infected.</h4>
              <form action = "http://localhost:5000/" id="upload-file"
method="post" enctype="multipart/form-data">
                    <label for="imageUpload" class="upload-label">
                         Choose...
                    </label>
                    <input type="file" name="image" id="imageUpload"
accept=".png, .jpg, .jpeg">
```

```
                    </form>
                    <div class="image-section" style="display:none;">
                        <div class="img-preview">
                            <div id="imagePreview">
                            </div>
                        </div>
                        <div>
<button type="button" class="btn btn-info btn-lg " id="btn-predict">Predict!</button>
                        </div>
                    </div>
<div class="loader" style="display:none;"></div>
<h3>
                        <span id="result"> </span>
</h3>
</div></div></div>
                </div>
                </div>
   </div>
</body>

<footer>
   <script src="{{ url_for('static', filename='js/main.js') }}" type="text/javascript"></script>
</footer>

</html>
```

Testing the code in IBM Watson Studio

Testing the model

#Replace the credentials that you got from Watson Machine Learning service

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
```

```
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey":"F_EEeP_U35v3wOqRVLUrBrCyI2-aSL91RZs2Lk4jfGZE"
}
client= APIClient (wml_credentials)

def guid_from_space_name (client, space_name):
    space = client.spaces.get_details()
    #print (space)
    return(next(item for item in space['resources'] if item['entity']["name"] == space_name) ['
metadata']['id'])

space_uid=guid_from_space_name (client,'Pneumonia_Detection')
print("Space UID = "+ space_uid)
Space UID = eb5d78cf-ec9b-4057-bdb0-cc07e107d466

client.set.default_space(space_uid)
'SUCCESS'

client.repository.download("8ad7e9bc-d558-4bbe-8720-c9e675ffd8ae",'Pneumo.h5.tar.gz')

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model("Pneumo.h5")
img=image.load_img(r"D:\Pneumonia Detection\chest_xray\train\PNEUMONIA\person12_b
acteria_47.jpeg",target_size=(128,128))
x=image.img_to_array(img)
print(x.shape)
(128, 128, 3)

import numpy as np
import pandas as pd
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
print(x.shape)
```

```
predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis=1)
classes_x
```

```
(1, 128, 128, 3)
```

```
array([1], dtype=int64)
index=["Normal","Pneumonic"]
result=str(index[classes_x[0]])
result
'Pneumonic'
```