

**C O V E N T R Y
U N I V E R S I T Y**

Faculty of Engineering, Environment and Computing

School of Computing, Electronics and Mathematics

MSc Data Science

7150CEM - Data Science Project

Prediction and Classification of Pneumonia using Machine Learning

Author: Harni Ramakrishnan

SID: 12141340

Supervisor: Xingang Wang

Submitted in partial fulfilment of the requirements for the Degree of Master of Science in Data Science

Academic Year: 2022/23

Declaration of Originality

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (<https://ethics.coventry.ac.uk/>) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: Harni Ramakrishnan

Date: 09/12/2022

First Name:	Harni
Last Name:	Ramakrishnan
Student ID number	12141340
Ethics Application Number	P143068
1 st Supervisor Name	Xingang Wang
2 nd Supervisor Name	Tabassom Sedighi

1 Abstract

Around 700,000 young children are lost to pneumonia-related causes every year, and the disease affects seven percent of the world's population. X-rays of the chest are often the first step in making a diagnosis for this condition. There is a demanding requirement to enhance the diagnostic precision. An effective model for the prediction of pneumonia based on digital chest X-ray pictures is provided in this study. If implemented, the model has the potential to assist radiologists in the process of decision making. Pneumonia is an illness that leads to inflammation of the lungs and may be fatal if it is not diagnosed and treated in a timely manner. Pneumonia is often diagnosed with a chest X-ray, which demands for an expert to examine the X-rays extensively. Expert examination of chest X-ray images to diagnose pneumonia is a time-consuming and inaccurate procedure. In this research, we present many variants of the TensorFlow Keras model for analyzing chest X-ray pictures for signs of pneumonia. Here, we describe the results of training the proposed TensorFlow Keras on both the original and the enlarged datasets to assess the impact of dataset size on TensorFlow Keras performance. This research created a low-cost diagnostic tool that identifies a chest x-ray image as pertaining to the normal or pneumonia division. The purpose of this study was to assist health diagnostics and professionals in the field of healthcare system. Results of the study yielded a high level of accuracy based on the proportion of correct responses. Displaying the confusion matrix of the system is an additional step in the validation process. The completed and suggested TensorFlow Keras model is able to attain an accuracy level of 93% under testing conditions.

Table of Contents

1	Abstract.....	3
	Table of Contents	4
	Acknowledgements	6
2	Introduction.....	7
	Background to the Project	7
	Project Objectives.....	8
	Client and User Requirements	8
	Overview of This Report	9
3	Literature Review	10
4	Methodology	16
	Training and Testing Data.....	17
5	Requirements.....	21
6	Analysis	21
7	Design.....	22
8	Implementation.....	23
	Language and Platform.....	23
	Python for Machine Learning	23
	Jupyter Notebook.....	24
	Handling the Data	25
	Training the model	25
	Model Building:	26
	Epoch.....	26
9	Results	31
	Need for Confusion Matrix:	33
10	Project Management	34
	Project Schedule	34
	Risk Management	35
	Quality Management	35
	Social, Legal, Ethical and Professional Considerations	36
11	Critical Appraisal	37
12	Conclusions.....	38
	Achievements	38

Future Work	38
13 Student Reflections	39
Bibliography and References	40
Appendix A – Interim Progress Report and Meeting Records.....	1
Appendix B – Certificate of Ethics Approval.....	2
Appendix C – Dataset and Code.....	3

Acknowledgements

I would like to express my wholehearted and sincere gratitude to all individuals who facilitated and supported me in doing this research project by their involvement, encouragement, and active participation.

From the early stages to the accomplishment of the study, Dr. Xingang Wang was tremendously helpful. His time and effort in this research provide new perspectives and ideas.

I would like to express my sincere thanks to Coventry University and to my course supervisor for granting me permission to carry out this study.

2 Introduction

Pneumonia may be caused by a bacterial, viral, or fungal infection of either lung, or both. Pus and other fluid fill the air sacs, indicating a severe infection. Anyone can get pneumonia, but very young children and older people are more likely to get it and it can be more dangerous for them. A cough, trouble breathing, a high temperature, and pain in the chest are other symptoms of pneumonia. It can also be caused by other infections or things like food or vomit going into your lungs.

In order to determine whether a patient has pneumonia, we ought to run a dataset consists of chest X-ray images. In this data collection, we have X-rays of the lungs from people with normal health, bacterial pneumonia, and viral pneumonia. In order to evaluate whether a patient has pneumonia or not, we are building a machine learning algorithm. To process the dataset, we intend to use the TensorFlow framework.

Background to the Project

Pneumonia is a dangerous infection that may be caused by viruses like those that cause the common cold and flu. This is responsible for around one third of all occurrences of pneumonia each year. When the virus penetrates your body, it causes your lungs to swell, which blocks oxygen from entering your bloodstream and causes you to get sick. The virus is able to disseminate through the atmosphere through a range of sources. It is most often transmitted by coughing, sneezing, or striking fluids of an infected surface.

Bacterial pneumonia may damage just a small portion of lung, or it might extend to the whole lung and infect the entire organ. If we have pneumonia, it is difficult for our lungs to take in the necessary amount of oxygen, which might potentially affect the regular operation of our cells.

The degree of pneumonia that is brought on by bacteria might vary. There are a variety of elements that might play a role in determining the severity of your pneumonia. The vitality of microorganisms the speed with which a diagnosis is made and therapy is administered to someone when they were born, how old are they, or how old they are right now.

The term "community-acquired pneumonia" refers to the kind of bacterial pneumonia that occurs most often in communities (CAP). An infection with microorganisms that was obtained by methods other than medical treatment is the root cause of aseptic meningitis. Inhalation of respiratory droplets from an infected individual who is coughing or sneezing, as well as direct

skin contact, are both effective means of transmitting the highly contagious airborne pneumococcal illness known as CAP.

Within three days of being exposed to infectious organisms at a medical institution, such as a hospital or a doctor's office, a patient runs the risk of developing hospital-acquired pneumonia, also known as HAP. This kind of pneumonia, which is commonly referred to as a "nosocomial infection," is sometimes more resistant to treatments and presents a greater challenge to cure than CAP does.

The bacteria known as streptococcus pneumoniae are the ones responsible for the vast majority of cases of bacterial pneumonia. The pulmonary system may be accessed by inhalation as well as intravenous administration of the substance. There is a vaccination available to protect against this strain.

Haemophilus influenzae is the microorganism responsible for the second highest number of cases of pneumonia. It is possible that this organism lives in the pharynx and nasal cavities of human hosts. This shouldn't do any damage or disease unless our immune system is already impaired to some extent.

Project Objectives

The chest x-rays in the dataset represent normal, bacterial pneumonia, and viral pneumonia. Developing a machine in order to determine whether the patient has pneumonia or not.

- The purpose of this research is to enhance the accuracy with which pneumonia may be predicted and classified using lung images of a patient that have been retrieved from the Kaggle dataset.
- Overcome this challenge by using the TensorFlow Keraslibrary.
- Outline a strategy for the model's training, validation, and testing procedures.
- Analyze how well the model is functioning.
- Recognize that future improvements and developments will be made to the solution.

Client and User Requirements

- Detect the images of respiratory x-rays
- Conduct a test to determine whether or not the patient has pneumonia.
- A camera with a good resolution is necessary for all of them.

Overview of This Report

This paper was written in order to contribute to the project entitled "Prediction and Classification of Pneumonia Using Machine Learning." During the research and development phases of the project, a number of significant issues were looked at and developed further; they are all included in the study. The majority of the discussion focuses on aspects which include the working principles, literature reviews, implementations, techniques, and future development, among other.

The study begins with an overview of the project's background, which covers every aspect of the project's objectives as well as its rationale. In addition to this, it outlines the project's user requirements, aims, and objectives. The desired outcomes and the actions that ought to be completed in order to achieve were described in the same section. The following section is the literature review, which consists of several studies, past experiences with similar risks, and the consequences of those interactions. Likewise, these were also scrutinized.

Following the literature review, the technique parts were covered, which included the algorithms and methods taken to address the issue, such as the kind of neural network deployed, and so on. Both data collection and data set generation are components of the same process. In the required section, both functional and non-functional components have been addressed. Later, a following part was added to the study that investigated numerous issues, methodology, and other crucial factors, such as the client's impact. Design addresses both the design architecture of the algorithm and the user interface of the result. The implementation section covers all the information of how each minor task was accomplished. It illustrates, for instance, how the command was developed and its basic structure. In the conclusion part, the ultimate key project outcome is displayed.

During the process of the project's execution, any prospective management issues, obstacles, or considerations will be discussed in the section related to project management. Along with that, there is a section on student's reflection that has been offered for summarizing what has been learned and accomplished as a result of the research. This section addresses the achievements and future work aspect of the conclusion, which is covered here.

In later stages, several sections for the appendix, which covers a variety of reports, were also included in the document.

3 Literature Review

(Enes Ayan & Halil Murat Ünver, 2019) states that pneumonia is an infectious lung disease that is caused by a bacterial infection. Early diagnosis is a vital part of delivering effective medication during any stage of the disease. In most cases, chest X-ray scans may provide adequate information for a knowledgeable and skilled radiologist to identify the illness. The diagnosis may be subjective for a wide range of reasons, including the fact that the appearance of the illness on chest X-ray images might be camouflaged or confused with that of other conditions. Therefore, technologies that provide doctors with aid via computer-assisted diagnostics are essential. Xception and Vgg16 are both well-known convolutional neural network models, and they used both of them to detect pneumonia in this research topic. During the training phase, they utilised both transfer learning and fine-tuning to improve the performance. According to the outcomes of the tests, the Vgg16 network is more accurate than the Xception network, with 0.87% accuracy compared to 0.82% accuracy. On the other hand, the Xception network was much more successful in recognising instances of pneumonia. As a consequence of this, they gained an insight that each network has its own set of skills when applied to the same dataset. In this specific study, a comparison of the effectiveness of two CNN networks in identifying patients who suffer from pneumonia was carried out. Transfer learning and fine-tuning were two methods that were used during the training of the models. Following the training phase's completion, they've compared the results of two separate network test cases. The outcomes of the tests clearly indicates that the Vgg16 network is superior than the Xception network in terms of accuracy by 0.87 percentage points and in terms of normal precision by 0.86 percentage points. Each network has its own inherent detection skills on the dataset, which are determined by the results of experimental research and confusion matrices. When it comes to identifying patients with pneumonia, the Xception network performs much better than the Vgg16 network. In addition, the Vgg16 network is superior in terms of its ability to recognise regular instances. In the next phase of the effort, they must merge the two networks. They have a higher rate of success in diagnosing pneumonia based on chest X-ray images if researchers combine the advantages offered by the two networks.

According to Rachna Jain, et al., research from 2020, pneumonia impacts the lives of a substantial proportion of young folks across the globe each year. In 2016, there were an estimated 1.2 million instances of pneumonia diagnosed in children less than 5 years old, with 880,000 fatalities resulting from the disease. Pneumonia is one of the top causes of mortality in children and has a high incidence rate in regions such as South Asia and Sub-Saharan Africa.

Pneumonia is one of the top 10 leading causes of death worldwide, even in such industrialized nations as the United States. Early detection and treatment has the potential to significantly decrease the number of children who pass away from pneumonia in countries where the disease is common. As a consequence of this finding, the researchers developed Convolutional Neural Network models for the purpose of diagnosing pneumonia based on x-ray images. Multiple Convolutional Neural Networks were trained to classify x-ray images into one of two categories—pneumonia and non-pneumonia—through the use of various parameters, hyper parameters, and the number of convolutional layers. This was accomplished by adjusting a variety of parameters and hyper parameters. The article covers a total of six different models. When compared to the second model, the first model simply contains two convolutional layers. The second model, conversely, has three. All of the other models, VGG16, VGG19, ResNet50, and Inception-v3, already had their training done. The equivalent precisions for VGG16, VGG19, ResNet50, and Inception-v3 are 87.28 %, 88.46 %, 77.56 %, and 70.99 %, respectively. This research study demonstrates that there are two neural networks that have excellent performance when used to real-time applications. Both varieties are notable for their pinpoint accuracy and unwavering reliability. Because it is essential to reduce dimensionality consequences in the context of medical imaging, recall is an essential performance parameter. Medical professionals are able to make efficient use of the Model 2 and VGG19 models for the early detection of pneumonia in both children and adults as a result of their remarkable performance across all assessment criteria. It is possible to do a quick analysis of a large number of x-ray pictures in order to provide highly accurate diagnostic findings. This helps healthcare systems provide more effective treatment for patients and reduce death rates. The authors of this paper plan to continue their study in the future with the goal of improving the overall classification accuracy of all models via the process of fine-tuning each parameter and hyper parameter. Utilizing datasets with a greater number of observations is one way to enhance the overall efficiency of the models.

Pneumonia is a respiratory infection that can be caused by bacteria or viruses, as explained by Rohit Kundu et al., 2021; it affects a large number of people, particularly in developing and underdeveloped nations where pollution, unhygienic living conditions, overcrowding, and inadequate medical infrastructure are prevalent. Pleural effusion is a condition that may develop as a consequence of pneumonia. In this condition, fluids fill the lung and make it difficult to breathe. It is essential to diagnose pneumonia at an early stage so that curative treatment may be delivered and the patient's chances of survival are increased. Imaging using chest X-rays is the most frequent method for detecting pneumonia. Nevertheless, examining chest X-rays is a difficult endeavor that is prone to subjective fluctuation. During the course of this research, we

developed a computer-aided diagnostic system with the purpose of automatically identifying cases of pneumonia based on chest X-ray images. We constructed an ensemble of three convolutional neural network models utilising deep transfer learning to handle the scarcity of available data. These models are GoogLeNet, ResNet-18, and DenseNet-121. In order to determine the weights that ought to be granted to the base learners in a weighted average ensemble methodology, an innovative method was used. The weight vector was frequently set experimentally in the scientific literature, which is a process that is prone to inaccuracy. The recommended approach was tested using two different publicly accessible pneumonia X-ray datasets, one of which had been supplied by Kermamy et al., while the other was provided by the Radiological Society of North America (RSNA). The testing was done using a five-fold cross-validation procedure. The results were superior to those that were achieved using state-of-the-art procedures, and our method outperformed ensemble techniques that are often used in context. The robustness of the method was proven statistically by utilising McNemar's and ANOVA tests on the datasets.

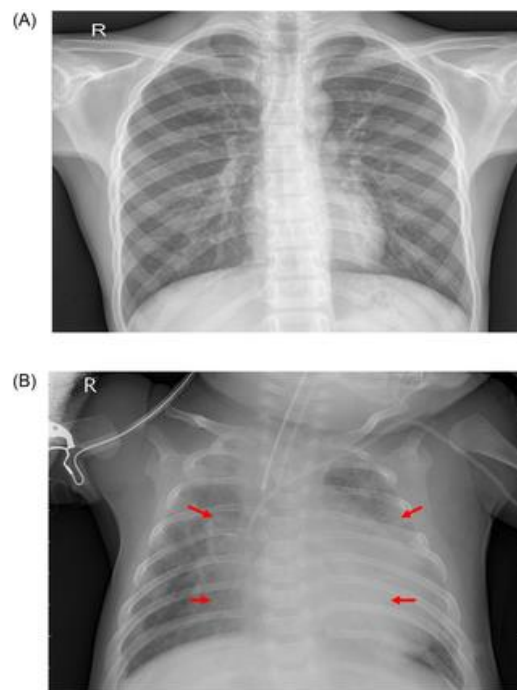


Fig 1. Comparative X-Ray Image of Healthy Individual and Person with Pneumonia

Pneumonia is a potentially life-threatening illness, therefore it's essential to diagnose it as early as possible so that the patient may get the appropriate care. Although chest radiographs are the most widely used tool for diagnosing pneumonia, they are subject to inter-class variability, and a proper diagnosis relies on the clinicians' ability to recognize early symptoms of the disease. In this research, they developed a computer-aided diagnosis (CAD) system that uses a

categorization method based on deep transfer learning to automatically sort chest X-ray images into "Pneumonia" and "Normal" categories.

According to the World Health Organization, problems with one's respiratory system are among the primary reasons why people died. Any malfunction of the pulmonary system is considered to be a respiratory illness when seen through the lens of medical terminology. These disorders may be broken down into two basic categories, which are known as obstructive and restrictive, according to the pattern of the sickness. Restrictive lung disease is defined by increased lung stiffness and limitation of lung expansion, while obstructive lung disease is distinguished by the blockage of airways. Both of these conditions make it challenging to breathe. Both of the above lung illnesses may be devastating if they are not appropriately diagnosed and treated in the earlier stages. Pneumonia is an inflammatory lung disease that claims the lives of an estimated four million people every year, the vast majority of whom are children (approximately 75 %). It is explicit that it is widespread in less industrialised regions, such as the solitary parts of Asia and Africa south of the Sahara desert. One of the primary challenges associated with the diagnosis of pneumonia is the tendency of medical professionals to over diagnose the condition. On occasion, medical professionals may show up diagnosis without taking the typical indications or symptoms into consideration. Several studies on the detection of bronchial infections in their preliminary stages have been carried out. The first objective is to improve the accuracy of its forecasts by the implementation of an innovative evolutionary-based fuzzy cognitive map (FCM) technology. The assertions state that the outcomes were more effective in predicting the outcomes of the researched medical records than the predictions that were created utilising the conventional genetic-based algorithm techniques.

Activation maps were employed in this study, and they assisted the researchers in locating portions of the picture that were most symptomatic of pneumonia. After completing all of the networks' convolutional layers, the final layer's activation maps were produced. In the instance of bacterial pneumonia, shown in Figure 2, every single network was successful in detecting the aberrant lung and accurately predicting the existence of pneumonia. In both lungs, the patient had a more diffuse "interstitial" pattern, which was indicative of viral pneumonia and was picked up by all of the fine-tuned designs [59]. (Figure 3).

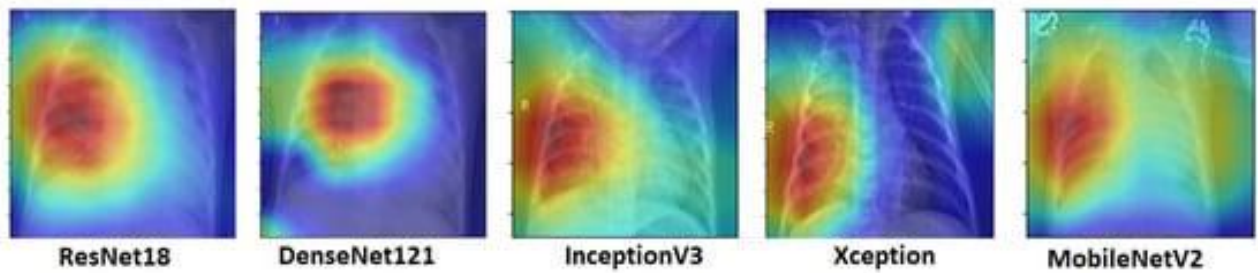


Fig 2. Activation maps for chest X-rays showing bacterial pneumonia that match to various architectural configurations. The deep learning models were able to recognize abnormal lungs in the case of bacterial pneumonia.



Fig 3. Activation maps for chest X-rays that show viral pneumonia that match to various architectural patterns. Deep learning models were able to determine viral pneumonia with a more dispersed "interstitial" pattern in both lungs. This pattern was found in both lungs.

Pneumonia is the medical term for a severe acute respiratory illness that damages the lungs. It is lethal because the air sacs get loaded with fluids and pus. There are essentially two types of pneumonia, bacterial and viral. Both may cause pneumonia. In most cases, the symptoms of bacterial pneumonia are more severe than those of viral pneumonia. The primary difference between viral and bacterial pneumonia is the approach that is used to treat the illness. In most cases, patients with viral pneumonia recover without any medical intervention, but those with bacterial pneumonia need antibiotic treatment. The illness has a significant presence in every region of the planet. The presence of high amounts of pollution is one of the primary causes of this. In many regions of the globe, there is a scarcity of skilled medical professionals and radiologists, which is problematic since their diagnoses and prognoses of illnesses of this kind are of critical importance. These days, more and more people are gravitating toward using computer-aided diagnostic tools that are powered by artificial intelligence. This facility is easily accessible to a huge population at a reasonable price. Because the symptoms of this disease often correlate with those of other illnesses, it may be difficult for radiologists to appropriately identify this ailment. This is one of the problems with having this condition. All of these concerns are addressed by deep learning algorithms, and their illness prediction accuracy is comparable to or, in some cases, even superior than that of a standard radiologist. The present generation is

accountable for the development that leads to a more enlightened future. The technological progress that has been made in recent times gets us one step closer to achieving human intelligence. Deep learning now has the ability to replicate the way that learning occurs in the human brain. It provides a solution to problems that are encountered on a regular basis in everyday life. Deep learning relies on convolutional neural networks because of its ability to obtain significant features for picture classification jobs and give medically promising results for image analysis. CNN Advantages is able to assist in the identification of certain characteristics inside an image and then utilize these features to generate probabilities for classifying certain input. This research made a significant contribution by developing a CNN deep-learning model that is optimised for reliably identifying and classifying pneumonia infections.

4 Methodology

In this section, we provide an overview of the exhaustive testing and assessment techniques that were utilised to establish whether the proposed paradigm is successful or not. This research was carried out using a chest X-ray image collection that was authorized. We used the open-source Keras deep learning framework in accordance with a TensorFlow backend in order to develop and train the convolutional neural network model. The categorization will be carried out by using the TensorFlow and Keras platforms respectively.

Dataset:

We've extracted the chest x-ray dataset from the Kaggle website. The dataset includes images of paediatric patients ranging in age from one to five years old, who were all seen at the Guangzhou women and children's medical Centre in Guangzhou. There are three distinct folders included in the dataset; these are the training, testing, and validation folders.

They have a total of 5856 x-ray images present in the dataset, split into two subfolders labelled "pneumonia" (P) and "normal" (N). There are a total of 1583 normal x-ray images and 4273 pneumonia x-ray images included in the collection.

Before doing an examination of the images obtained from chest x-rays, each individual image was subjected to an initial screening for quality control. This would include eradicating the certain scans that were of poor quality or could not be read. Two subsets, namely training and testing, were created based on dataset after it was partitioned.

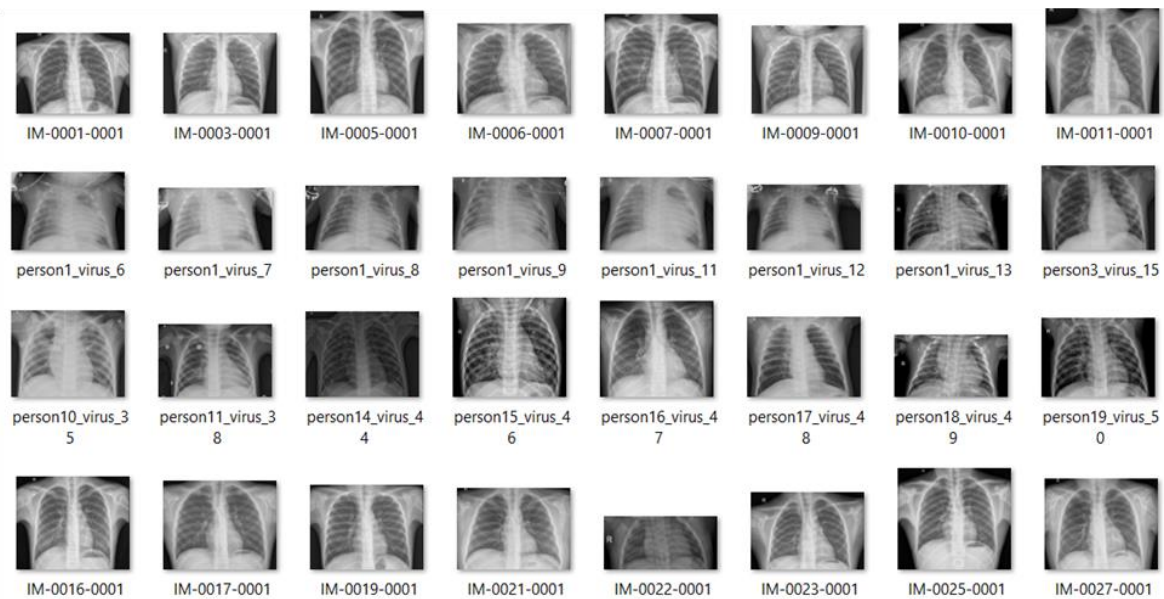


Fig 4: Some illustrations from the dataset

Training and Testing Data

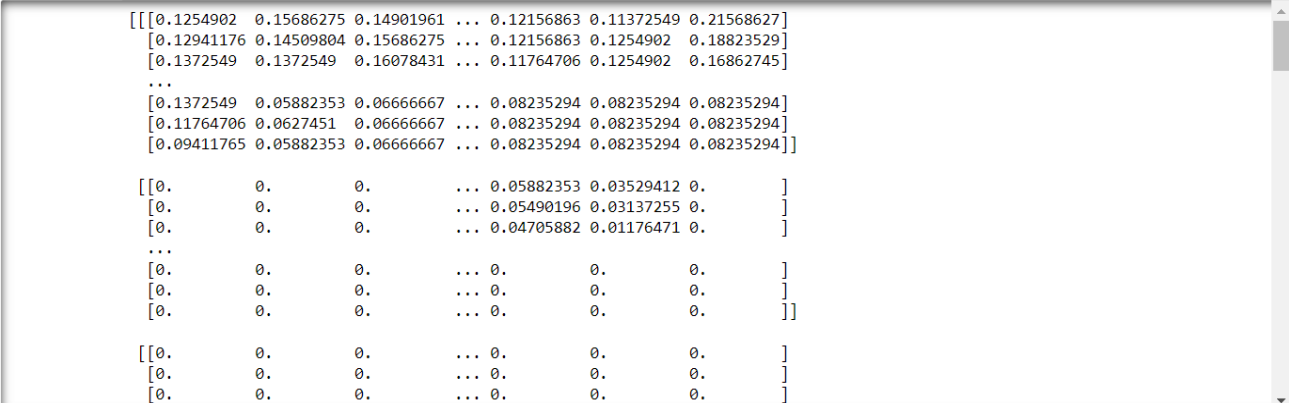
In order to facilitate the preliminary processing of the data, the data have been segregated into training and testing. 70% of the data were used for training purpose, while just 30% were used for testing. The Python code that was used to separate the data for testing and training is shown in Figure 5.

```
In [16]: X = x_train + x_test
         y = y_train + y_test

In [17]: x_train, x_test, y_train, y_test = train_test_split( X, y, test_size=0.3, random_state=42)

In [18]: # Normalize the data
         x_train = np.array(x_train) / 255
         x_val = np.array(x_val) / 255
         x_test = np.array(x_test) / 255

In [19]: print(x_train)
         print(x_val)
         print(x_test)
```



```
[[[0.1254902  0.15686275 0.14901961 ... 0.12156863 0.11372549 0.21568627]
 [0.12941176 0.14509804 0.15686275 ... 0.12156863 0.1254902  0.18823529]
 [0.1372549  0.1372549  0.16078431 ... 0.11764706 0.1254902  0.16862745]
 ...
 [0.1372549  0.05882353 0.06666667 ... 0.08235294 0.08235294 0.08235294]
 [0.11764706 0.0627451  0.06666667 ... 0.08235294 0.08235294 0.08235294]
 [0.09411765 0.05882353 0.06666667 ... 0.08235294 0.08235294 0.08235294]]

[[[0.      0.      0.      ... 0.05882353 0.03529412 0.      ]
 [0.      0.      0.      ... 0.05490196 0.03137255 0.      ]
 [0.      0.      0.      ... 0.04705882 0.01176471 0.      ]
 ...
 [0.      0.      0.      ... 0.      0.      0.      ]
 [0.      0.      0.      ... 0.      0.      0.      ]
 [0.      0.      0.      ... 0.      0.      0.      ]]]

[[[0.      0.      0.      ... 0.      0.      0.      ]
 [0.      0.      0.      ... 0.      0.      0.      ]
 [0.      0.      0.      ... 0.      0.      0.      ]]]
```

Fig 5. Data isolation for training and testing

Models

The most important component of this study was selecting a model that was suitable for the requirements of the project. According to the research that were looked at, the CNN algorithm continues to be the model that is the most often used and that performs well in image classification methods.

TensorFlow provides access to both high and low level application programming interfaces (APIs), while the Keras library solely provides access to high level APIs. Since Keras is written in Python, it is simpler to use than TensorFlow.

➤ TensorFlow

TensorFlow is an open-source library for numerical calculation that is compatible with the Python programming language. It streamlines the process of machine learning and makes it simpler to create neural networks. TensorFlow, which competes with frameworks such as PyTorch and Apache MX Net, has the ability to train and run deep neural networks. These networks can be used for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation)-based simulations. In addition, TensorFlow can be used for PDE-based simulations. The best part is that TensorFlow allows for production prediction at scale, making use of the same models that are used during training.

In addition to this, TensorFlow contains a huge library of pre-trained models that we can include into our own projects. When we are training our own models, we can also utilise the code that is provided in the TensorFlow Model Garden as an example of evidence - based practices. With order to handle the image dataset, we are putting in action with Keras package that is included in TensorFlow. The structure of the TensorFlow block diagram's hierarchy is shown in the following figure 6.

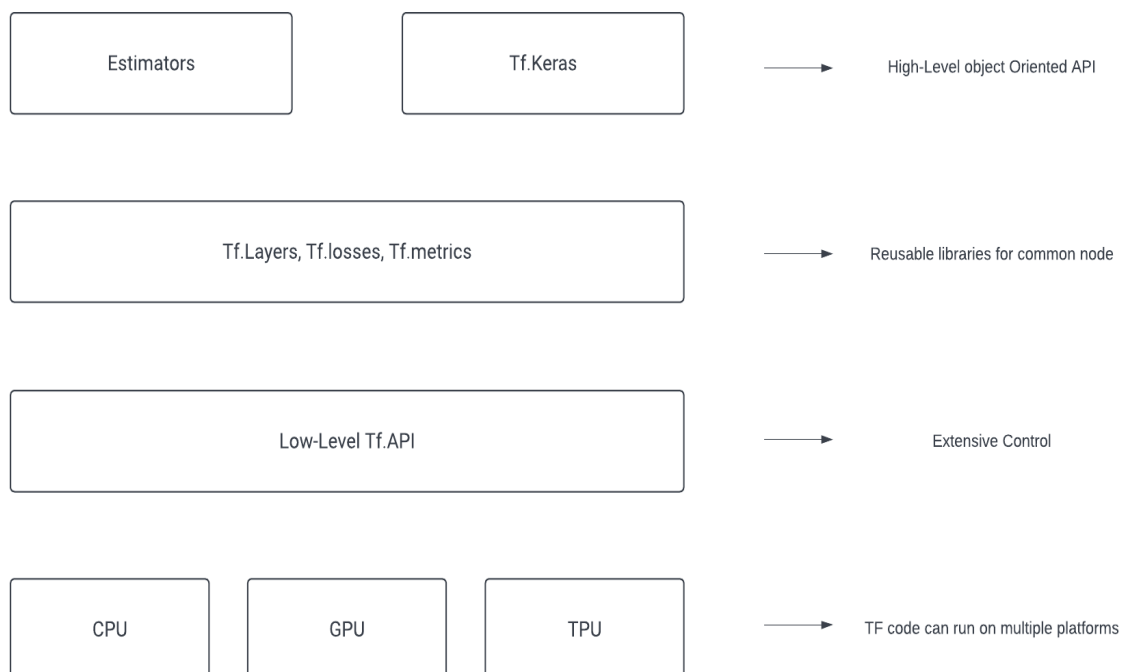


Fig 6. TensorFlow Block Diagram.

➤ Keras

Keras is an Application Programming Interface (API) for neural networks that is written in Python. It is strongly connected with TensorFlow, which is a library that is used in the construction of machine learning models. Keras models provide a straightforward and user-friendly method of defining a neural network, which is subsequently carried out by TensorFlow on the user's behalf. On the other hand, Keras is a high-level application programming interface that operates on top of TensorFlow. Keras simplifies the implementation of complex neural networks with its easy-to-use framework.

A deep learning API written in Python, Keras is built on top of Google's TensorFlow library for machine learning. It was designed with the intention of making significant experimentation simpler to accomplish. Being capable of moving promptly from the design phase to the analysis and interpretation phase is inherent to conduct qualitative research.

Keras is described as straightforward without seeming too naive. Keras reduces the high level of anxiety that is placed on developers, permitting them to concentrate better one's attention on a specific aspects of the issue.

In other words, individuals should be able to accomplish what might appear to be impossible mission by pursuing the progression outlined by Keras. Workflows that are simple have to be rapid and intuitive.

Keras is a powerful framework that delivers both performance and scalability that are on track with the finest in the industry; NASA, YouTube, and Waymo are just a handful of the companies that rely on it. The structure of TensorFlow Keras is shown in the figure that is displayed below (deep learning).

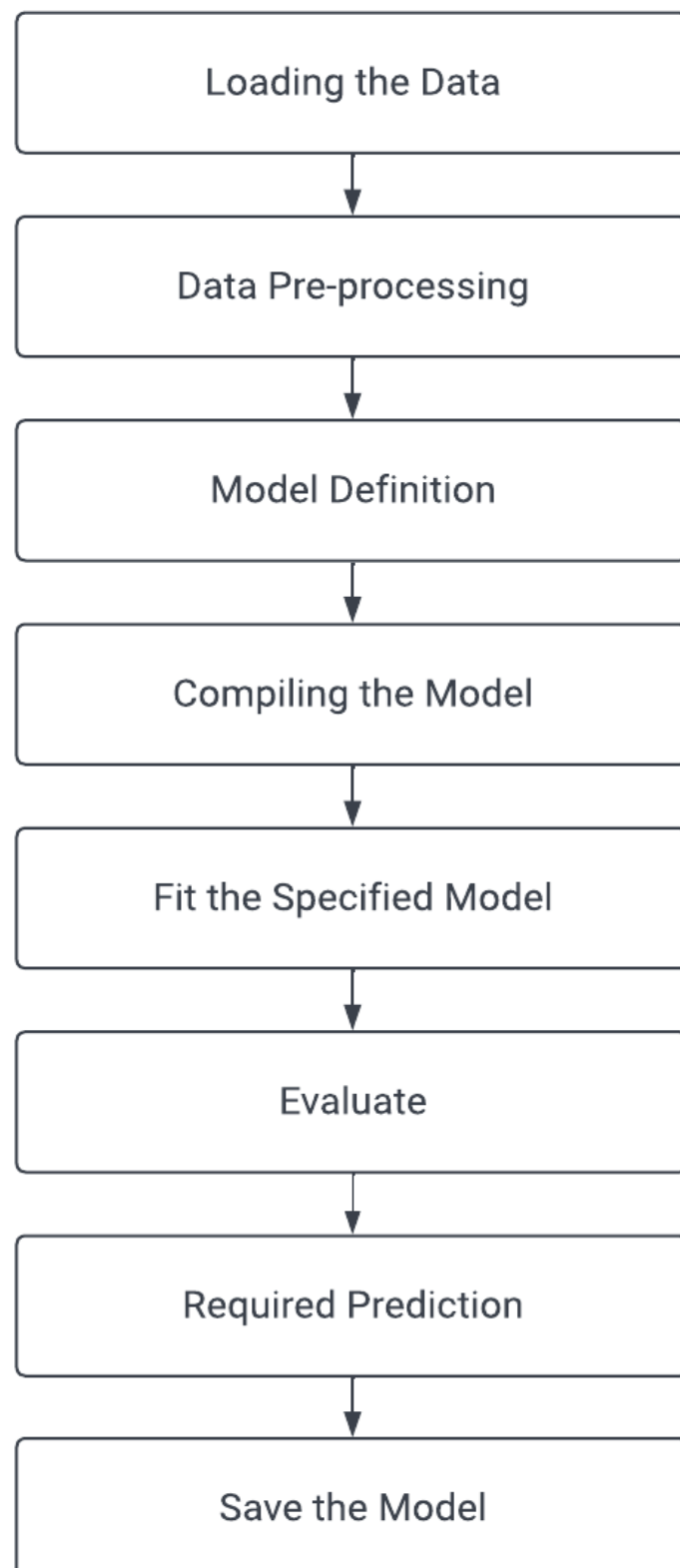


Fig 7. Block diagram of TensorFlow Keras (Deep Learning)

5 Requirements

The project incorporates the machine learning algorithms that are now running on the local computer. Python is the language that is used for programming, and Jupyter Notebook is the online code editor that is used. This project has been put through its paces on Windows 11. This project has the following represents the minimum criteria that may be placed on running this implementation:

1. CPU with an i5 processor
2. RAM: 8 GB
3. Storage: 512 GB

For the creation of this model, the following tools are required:

1. Jupyter Notebook
2. Programming Language: Python
4. Keras and TensorFlow are two of them.
5. The Process of Implementing Algorithms

6 Analysis

The purpose of the preprocessing step is to enhance the image's quality so that we can do a more thorough analysis of it. We are able to eliminate unwanted distortions and improve some characteristics that are essential for the specific application that we are working on as a result of the preprocessing that we do. It's possible that these characteristics may change depending on the requirements.

Pre-Processing:

When we use the vertical flip or the horizontal flip augmentation, it indicates that the pixels will be flipped either row-wise or column-wise.

we will be able to achieve this to a much greater extent, not only by inverting the photos vertically but also by adjusting the angle of the image by one degree. This will result in a total

of samples for each instance that is included in your training set. Depending on the speed of your algorithm, this may be a reasonably decent technique to ensure that the algorithm is not simply trained to detect images and their mirror counterparts. However, it is important to note that this method is dependent on the speed of your algorithm.

```
In [20]: # resize data for deep learning
x_train = x_train.reshape(-1, img_size, img_size, 1)
y_train = np.array(y_train)

x_val = x_val.reshape(-1, img_size, img_size, 1)
y_val = np.array(y_val)

x_test = x_test.reshape(-1, img_size, img_size, 1)
y_test = np.array(y_test)

In [21]: datagen = ImageDataGenerator(
    featurewise_center=False, # set input mean to 0 over the dataset
    samplewise_center=False, # set each sample mean to 0
    featurewise_std_normalization=False, # divide inputs by std of the dataset
    samplewise_std_normalization=False, # divide each input by its std
    zca_whitening=False, # apply ZCA whitening
    rotation_range = 30, # randomly rotate images in the range (degrees, 0 to 180)
    zoom_range = 0.2, # Randomly zoom image
    width_shift_range=0.1, # randomly shift images horizontally (fraction of total width)
    height_shift_range=0.1, # randomly shift images vertically (fraction of total height)
    horizontal_flip = True, # randomly flip images
    vertical_flip=False) # randomly flip images

    datagen.fit(x_train)
```

7 Design

The detection model was used to analyze the input X-ray images in order to determine whether the sample input X-ray had any symptoms of pneumonia or not. The input will undergo a preliminary processing step first. The goal of pre-processing is to enhance the image data by either minimizing the number of unintentional distortions or enhancing the quality of particular aspects that are essential for the processing that generally happens. After the image has been pre-processed, it is then run through the model. A model is a function that has been trained to detect certain types of patterns by being exposed to several sets of data and going through extensive training. In order to provide a prediction, the model is consumed with the images that are being offered. The figure above depicts the system architecture block diagram.

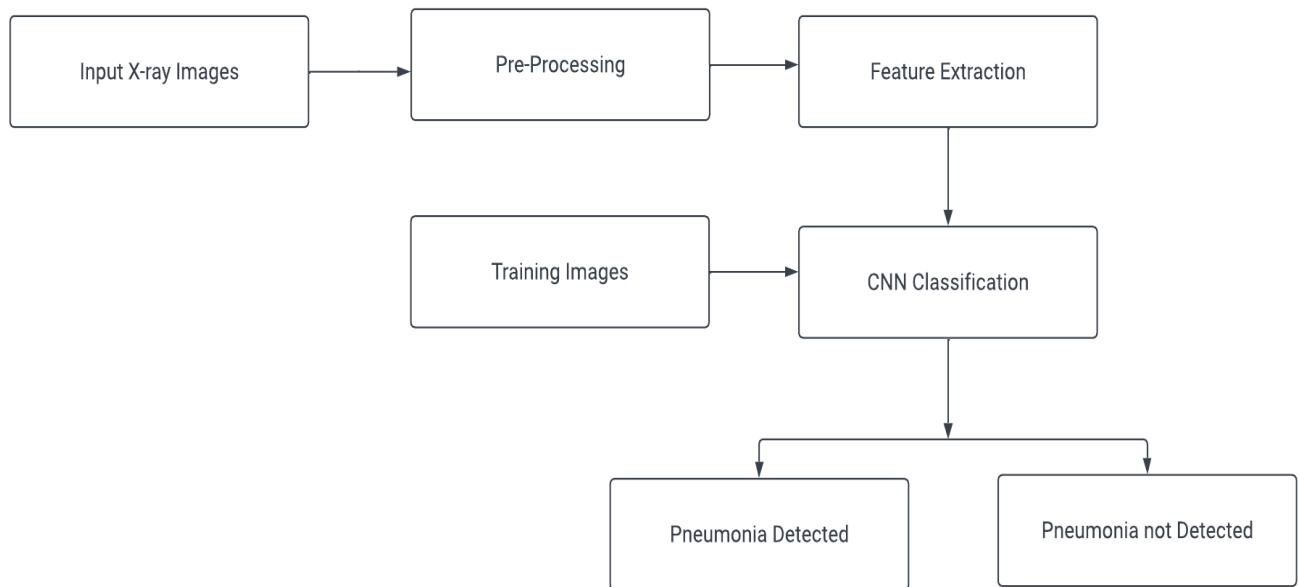


Fig 8. System Architecture.

8 Implementation

Language and Platform

Throughout this context, Python is the programming interface. In this case, the Jupyter Notebook is chosen as the code editor. The neural network and model implementation are both carried out in Python.

Python for Machine Learning

In order to complete this project, Jupyter Notebook was used. Python is used throughout the process of developing the neural network, as well as for the model's actual implementation. It was determined that the Python programming language is the best option for implementing the important technologies and methodologies that were discussed. This decision was made after considering the methodology, the necessary supporting documentation, and the availability of open source library software. The first version of the Python programming language was made available to the public in the early 1990s. Python has now developed to become one of the most popular programming languages owing to its many desirable characteristics, including the fact

that it is open source and free, that its code is readable, that it takes an object-oriented approach, portable, expressive, and so on.

The existence of algorithm libraries in Python, such as Keras, TensorFlow, Scikit-learn, Pandas, and Pytorch, is the key factor that contributes to their implementation in the process of resolving issues related to machine learning. There are additionally libraries for the graphical representation of the data, as well as libraries for the creation of graphs and charts to display that data.

Jupyter Notebook

Jupyter Notebook, previously known as IPython Notebook, is an interactive computing environment that can be accessed through the web and used to create notebook documents. Jupyter Notebook was developed with the help of a range of open-source libraries, the most prominent of which includes IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax. A Jupyter Notebook application is a REPL that runs in a browser and has an ordered list of input/output cells that may include code, text (using Markdown), mathematics, graphs, and rich media. These cells can also contain an ordered list of input/output cells. The Jupyter Notebook is a server-client application that allows editing and running notebook documents via a web browser.

The notebook interface of other applications like as Maple, Mathematica, and SageMath is comparable to the interface of Jupyter Notebook. This form of computational interface was pioneered by Mathematica in the 1980s.

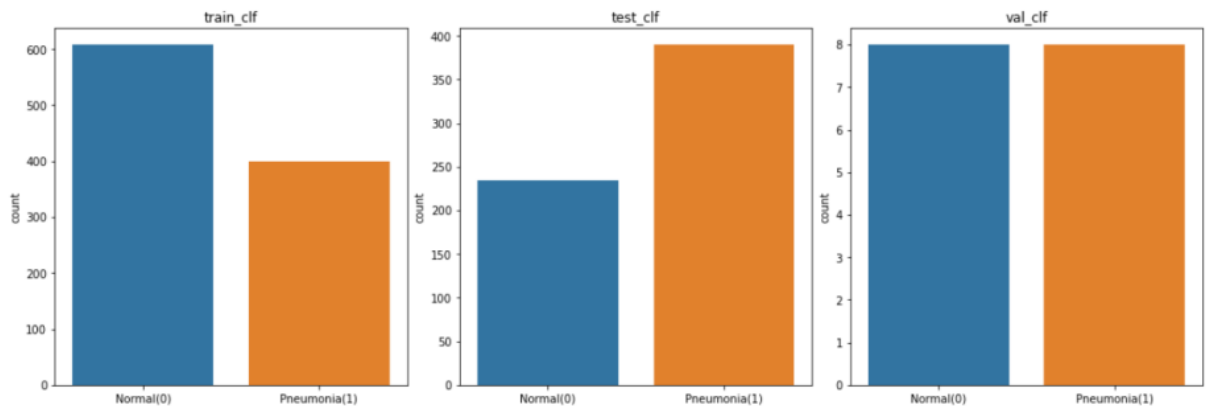
The term "Jupyter Notebook" may be used to refer to two entirely distinct ideas: either the user-facing programme that allows users to edit code and text, or the underlying file format that is compatible with several alternative implementations.

The Jupyter Notebook and other tools that are derivatives of it have been adopted by major cloud computing providers as the frontend interface for cloud customers. Jupyter supports over 40 programming languages.

Handling the Data

```
In [14]: fig, ax = plt.subplots(1,3,constrained_layout=True, figsize=(15, 5))
axesSub = sns.countplot(train_label,ax=ax[0])
axesSub.set_title('train_clf')
axesSub = sns.countplot(test_label, ax=ax[1])
axesSub.set_title('test_clf')
axesSub = sns.countplot(val_label, ax=ax[2])
axesSub.set_title('val_clf')
```

Out[14]: Text(0.5, 1.0, 'val_clf')



Training the model

In order to make the initial processing of the data simpler, the data have been separated into training and testing categories. Just 30% of the data were utilized for actual testing, while the remaining 70% were put to use for training purposes. The following is an example of the Python code that was used in order to divide the data for testing and training.

```
In [15]: x_train = []
y_train = []

x_val = []
y_val = []

x_test = []
y_test = []

for feature, label in train:
    x_train.append(feature)
    y_train.append(label)

for feature, label in test:
    x_test.append(feature)
    y_test.append(label)

for feature, label in val:
    x_val.append(feature)
    y_val.append(label)
```

```
In [16]: X = x_train + x_test
y = y_train + y_test
```

```
In [17]: x_train, x_test, y_train, y_test = train_test_split( X, y, test_size=0.3, random_state=42)
```

```
In [18]: # Normalize the data
x_train = np.array(x_train) / 255
x_val = np.array(x_val) / 255
x_test = np.array(x_test) / 255
```

```
In [19]: print(x_train)
         print(x_val)
         print(x_test)

[[[0.1254902  0.15686275 0.14901961 ... 0.12156863 0.11372549 0.21568627]
  [0.12941176 0.14509804 0.15686275 ... 0.12156863 0.1254902  0.18823529]
  [0.1372549  0.1372549  0.16078431 ... 0.11764706 0.1254902  0.16862745]
  ...
  [0.1372549  0.05882353 0.06666667 ... 0.08235294 0.08235294 0.08235294]
  [0.11764706 0.0627451  0.06666667 ... 0.08235294 0.08235294 0.08235294]
  [0.09411765 0.05882353 0.06666667 ... 0.08235294 0.08235294 0.08235294]]

[[[0.         0.         0.         ... 0.05882353 0.03529412 0.         ]
  [0.         0.         0.         ... 0.05490196 0.03137255 0.         ]
  [0.         0.         0.         ... 0.04705882 0.01176471 0.         ]
  ...
  [0.         0.         0.         ... 0.         0.         0.         ]
  [0.         0.         0.         ... 0.         0.         0.         ]
  [0.         0.         0.         ... 0.         0.         0.         ]]]

[[[0.         0.         0.         ... 0.         0.         0.         ]
  [0.         0.         0.         ... 0.         0.         0.         ]
  [0.         0.         0.         ... 0.         0.         0.         ]]]
```

Model Building:

Epoch

While all of the training data is utilized at the same time, this is referred to as an epoch. An epoch is defined as the total number of iterations that occur when using all of the training data in one cycle to train a machine learning model. An further definition of an epoch utilizes the number of times a training dataset is iterated through an algorithm.

After one epoch, each sample in the training dataset will have the chance to change the internal model parameters. This is what is meant by the term "one epoch." The number of batches that make up an era might range from one to numerous. An epoch that only contains one batch, for instance, is referred to as the batch gradient descent learning method in the previous case.

Consider a for-loop that spans the whole number of epochs, with each iteration of the loop moving forward through the training dataset. loop that iterates through each batch of samples, where one batch contains the stated "batch size" number of samples. This second for-loop is nested within the first for-loop.

Kares flatten

The Keras flatten command is a method for providing input in order to create an additional layer when flattening using the flatten class. The output of keras flatten is the same size as the input, and the batch size is unaffected. If the supplied input for the value is 2, then the anticipated output with keras flatten will turn out to be 4. This indicates that an additional layer and more parameters will need to be added in order to streamline the whole process. Its most common are

used in situations involving any of the multi-dimensional tensors that consist of image datasets and multi-layer datasets, and which do not permit the loss of any information from the respective datasets. The `flatten()` method returns a copy of the array once it has been flattened down to a single dimension.

One of the open-source and free machine learning-oriented application programming interfaces (APIs) that is used for simply developing complicated neural network architecture is the Keras library, which is an extension of TensorFlow. It provides assistance in training the models in a seamless manner, permitting the imports to the trained model to be simply handled by using `keras flatten`. Therefore, let's go right into the workings of neural network models that need input, followed by corresponding output, and discuss how to utilise them.

- The following step, which occurs once the essential libraries for keras flattened have been loaded, is to handle the `keras flatten` class.
- The input tensors should then be imported like picture datasets, with the exception that the input data should correspond to the appropriate input layer.
- In order to transform a multi-dimensional array into a one-dimensional flattened array, the `layer.flatten()` function is used. Another way to phrase this is to state that a single-dimensional array is created.
- The basic idea behind `keras.layer.flatten` is to treat the feed of the input as a multi-dimensional array and to treat the anticipated output as a single-dimensional array.

Keras Conv2D class

Keras `conv2D`, which stands for convolution layer in a 2-dimensional pattern, is the component that is in charge of generating the kernel of convolution. This kernel is then combined with the other input layers of the Keras model to produce the final resultant output, which will be in the form of a tensor.

Keras `conv2D` is the layer of convolution that helps us construct the kernel of convolution. This is necessary for the model to produce output containing tensor when it is connected with the input layers of the Keras model. The kernel that is created is a mask or matrix of convolution, and this matrix or mask is then utilised for edge detection, sharpening, blurring, and embossing, all of which are achieved by conducting convolution operations on images together with the kernel.

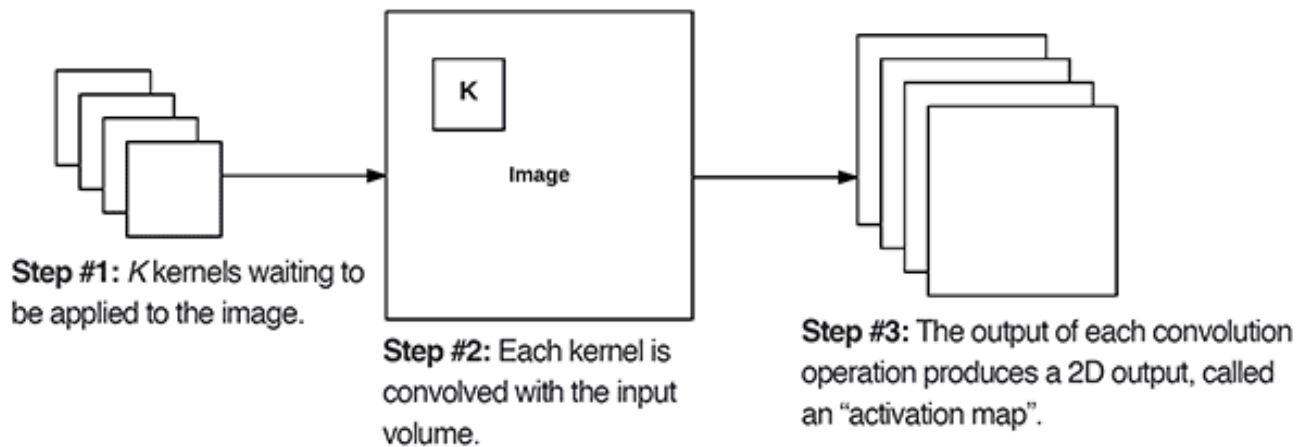


Fig . The Keras Conv2D parameter, `filters` determines the number of kernels to convolve with the input volume. Each of these operations produces a 2D activation map.

Keras - Dense Layer

The dense layer is the layer of a neural network that is regular and deeply interconnected. It is the layer that is utilized most often and is the most prevalent.

The dense layer is responsible for carrying out a matrix-vector multiplication in the background. Neural network architecture may be used to train and update the parameters, which are the values that are utilised in the matrix. Classifier can also be used to train new models.

An m -dimensional vector is what has been produced as a result of the thick layer being applied. As a result, the primary purpose of the thick layer is to alter the dimensions of the vector. In addition to this, dense layers perform operations on the vector, such as rotation, scaling, and translation.

MaxPooling2D layer

When applied to the spatial data, the max pooling procedures are carried out using the keras max pooling 2D layer. Importing the maxpooling2d library into the keras module is required before we can utilize the layer that maxpooling2d provides.

The pooling or max pooling process that Keras MaxPooling2D is responsible for calculating the maximum value present in each patch as well as the feature map. The results will be sampled, or it will pool features map that was emphasizing the most present feature into the patch that includes the average feature presence from the average pooling. Both of these options will result

in the same thing. It has been discovered that the max pooling works rather well in the average pooling for vision tasks.

```
In [22]: model = Sequential()
model.add(Conv2D(32, (3,3), strides = 1, padding = 'same', activation = 'relu', input_shape = (224,224,1)))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2), strides = 2, padding = 'same'))
model.add(Conv2D(64, (3,3), strides = 1, padding = 'same', activation = 'relu'))
model.add(Dropout(0.1))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2), strides = 2, padding = 'same'))
model.add(Conv2D(64, (3,3), strides = 1, padding = 'same', activation = 'relu'))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2), strides = 2, padding = 'same'))
model.add(Conv2D(128, (3,3), strides = 1, padding = 'same', activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2), strides = 2, padding = 'same'))
model.add(Conv2D(256, (3,3), strides = 1, padding = 'same', activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2), strides = 2, padding = 'same'))
model.add(Flatten())
model.add(Dense(units = 128, activation = 'relu'))
model.add(Dropout(0.2))
model.add(Dense(units = 1, activation = 'sigmoid'))
model.compile(optimizer = "rmsprop", loss = "binary_crossentropy", metrics = ['accuracy'])
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 224, 224, 32)	320
batch_normalization (Batch Normalization)	(None, 224, 224, 32)	128
max_pooling2d (MaxPooling2D)	(None, 112, 112, 32)	0
conv2d_1 (Conv2D)	(None, 112, 112, 64)	18496
dropout (Dropout)	(None, 112, 112, 64)	0
batch_normalization_1 (Batch Normalization)	(None, 112, 112, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 56, 56, 64)	0
conv2d_2 (Conv2D)	(None, 56, 56, 64)	36928
batch_normalization_2 (Batch Normalization)	(None, 56, 56, 64)	256
max_pooling2d_2 (MaxPooling2D)	(None, 28, 28, 64)	0
conv2d_3 (Conv2D)	(None, 28, 28, 128)	73856
dropout_1 (Dropout)	(None, 28, 28, 128)	0

batch_normalization_3 (Batch Normalization)	(None, 28, 28, 128)	512
max_pooling2d_3 (MaxPooling2D)	(None, 14, 14, 128)	0
conv2d_4 (Conv2D)	(None, 14, 14, 256)	295168
dropout_2 (Dropout)	(None, 14, 14, 256)	0
batch_normalization_4 (Batch Normalization)	(None, 14, 14, 256)	1024
max_pooling2d_4 (MaxPooling2D)	(None, 7, 7, 256)	0
flatten (Flatten)	(None, 12544)	0
dense (Dense)	(None, 128)	1605760
dropout_3 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

```

=====
Total params: 2,032,833
Trainable params: 2,031,745
Non-trainable params: 1,088

```

```

In [23]: learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 2, verbose=1, factor=0.3, min_lr=0.000001)

In [24]: history = model.fit(datagen.flow(x_train, y_train, batch_size = 16), epochs = 12, validation_data = datagen.flow(x_val, y_val),
Epoch 1/12
72/72 [=====] - 100s 1s/step - loss: 1.5620 - accuracy: 0.7723 - val_loss: 11.6284 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 2/12
72/72 [=====] - 82s 1s/step - loss: 0.5744 - accuracy: 0.8039 - val_loss: 15.0147 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 3/12
72/72 [=====] - ETA: 0s - loss: 0.4534 - accuracy: 0.8564
Epoch 3: ReduceLROnPlateau reducing learning rate to 0.0003000000142492354.
72/72 [=====] - 80s 1s/step - loss: 0.4534 - accuracy: 0.8564 - val_loss: 14.0534 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 4/12
72/72 [=====] - 84s 1s/step - loss: 0.2769 - accuracy: 0.8897 - val_loss: 16.8662 - val_accuracy: 0.5000 - lr: 3.0000e-04
Epoch 5/12
72/72 [=====] - ETA: 0s - loss: 0.2976 - accuracy: 0.9011
Epoch 5: ReduceLROnPlateau reducing learning rate to 9.000000427477062e-05.
72/72 [=====] - 85s 1s/step - loss: 0.2976 - accuracy: 0.9011 - val_loss: 17.0867 - val_accuracy: 0.5000 - lr: 3.0000e-04
Epoch 6/12
72/72 [=====] - 81s 1s/step - loss: 0.2519 - accuracy: 0.9037 - val_loss: 19.3704 - val_accuracy: 0.5000 - lr: 9.0000e-05
Epoch 7/12
72/72 [=====] - ETA: 0s - loss: 0.2513 - accuracy: 0.9221
Epoch 7: ReduceLROnPlateau reducing learning rate to 2.700000040931627e-05.
72/72 [=====] - 80s 1s/step - loss: 0.2513 - accuracy: 0.9221 - val_loss: 12.1980 - val_accuracy: 0.5000 - lr: 9.0000e-05
Epoch 8/12
72/72 [=====] - 79s 1s/step - loss: 0.2542 - accuracy: 0.9089 - val_loss: 8.5560 - val_accuracy: 0.5000

```

```

Epoch 9/12
72/72 [=====] - ETA: 0s - loss: 0.2149 - accuracy: 0.9221
Epoch 9: ReduceLROnPlateau reducing learning rate to 8.10000013655517e-06.
72/72 [=====] - 77s 1s/step - loss: 0.2149 - accuracy: 0.9221 - val_loss: 4.4751 - val_accuracy: 0.500
0 - lr: 2.7000e-05
Epoch 10/12
72/72 [=====] - 84s 1s/step - loss: 0.2261 - accuracy: 0.9168 - val_loss: 3.5274 - val_accuracy: 0.625
0 - lr: 8.1000e-06
Epoch 11/12
72/72 [=====] - 87s 1s/step - loss: 0.2467 - accuracy: 0.9151 - val_loss: 1.3527 - val_accuracy: 0.562
5 - lr: 8.1000e-06
Epoch 12/12
72/72 [=====] - ETA: 0s - loss: 0.2233 - accuracy: 0.9256
Epoch 12: ReduceLROnPlateau reducing learning rate to 2.429999949526973e-06.
72/72 [=====] - 82s 1s/step - loss: 0.2233 - accuracy: 0.9256 - val_loss: 1.0876 - val_accuracy: 0.562
5 - lr: 8.1000e-06

```

```

In [26]: print("Loss of the model is - ", model.evaluate(x_test,y_test)[0])
print("Accuracy of the model is - ", model.evaluate(x_test,y_test)[1]*100 , "%")

16/16 [=====] - 7s 384ms/step - loss: 0.2049 - accuracy: 0.9265
Loss of the model is - 0.20491719245910645
16/16 [=====] - 7s 400ms/step - loss: 0.2049 - accuracy: 0.9265
Accuracy of the model is - 92.65305995941162 %

```

9 Results

```

In [27]: epochs = [i for i in range(12)]
fig, ax = plt.subplots(1,2)
train_acc = history.history['accuracy']
train_loss = history.history['loss']
val_acc = history.history['val_accuracy']
val_loss = history.history['val_loss']
fig.set_size_inches(20,10)

ax[0].plot(epochs , train_acc , 'go-' , label = 'Training Accuracy')
ax[0].plot(epochs , val_acc , 'ro-' , label = 'Validation Accuracy')
ax[0].set_title('Training & Validation Accuracy')
ax[0].legend()
ax[0].set_xlabel("Epochs")
ax[0].set_ylabel("Accuracy")

ax[1].plot(epochs , train_loss , 'g-o' , label = 'Training Loss')
ax[1].plot(epochs , val_loss , 'r-o' , label = 'Validation Loss')
ax[1].set_title('Testing Accuracy & Loss')
ax[1].legend()
ax[1].set_xlabel("Epochs")
ax[1].set_ylabel("Training & Validation Loss")
plt.show()

```

Below are the graphs of accuracy and loss that we got for TensorFlow Keras method.

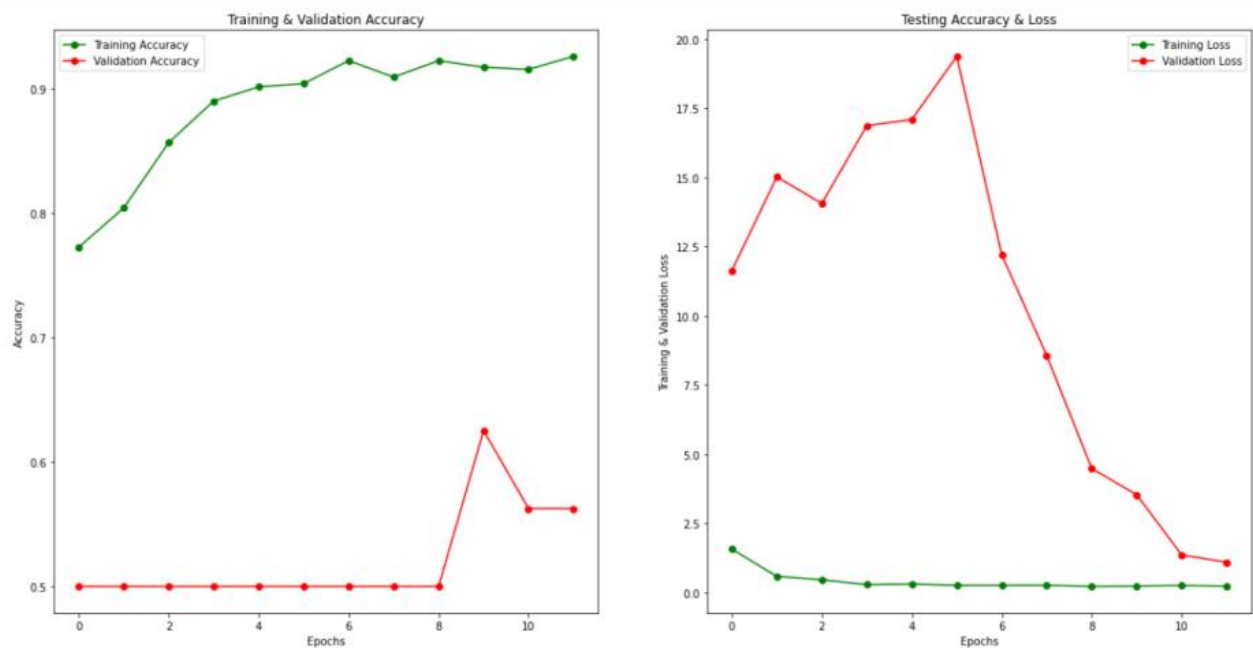


Fig . Graphs of Keras accuracy and loss

The assessment yielded a prediction and classification of PNEUMONIA with a percentage of approximately 93%.

```
In [29]: print(classification_report(y_test, predictions, target_names = ['Normal (Class 0)', 'Pneumonia (Class 1)']))
```

	precision	recall	f1-score	support
Normal (Class 0)	0.91	0.95	0.93	254
Pneumonia (Class 1)	0.95	0.90	0.92	236
accuracy			0.93	490
macro avg	0.93	0.93	0.93	490
weighted avg	0.93	0.93	0.93	490

Confusion Matrix

The confusion matrix is a kind of matrix that is used in the process of evaluating the efficiency of various classification models with regard to a certain collection of test data. If the actual values of the test data are known, then and only then can it be determined. Understanding the matrix self is not difficult, but the terminology that is used in conjunction with it could be confounding. As a result of the fact that it presents the mistakes in the model's performance in the form of a matrix, it is sometimes referred to as an error matrix.

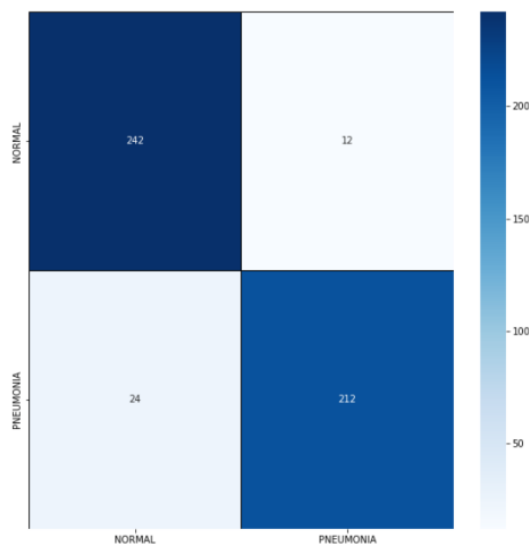
When analysing the efficacy of a classification model, a confusion matrix is an NxN matrix that is employed, where N is the number of target classes. The matrix makes a comparison between the real-world goal values and the values predicted by the machine learning model. This provides us with a comprehensive perspective of the performance of our classification model as well as the kind of mistakes that it is producing.

Need for Confusion Matrix:

- It demonstrates how any classification model will provide incorrect predictions when it does so.
- Not only does the confusion matrix provide insight into the mistakes that the classifier is making, but it also provides information into the sorts of errors that are being generated.
- This breakdown will assist you in overcoming the limitations that come with relying just on categorization accuracy.
- In the confusion matrix, each column reflects the number of occurrences of the corresponding predicted class.
- Every row in the confusion matrix depicts one instance of the class that is being confused with.
- It offers an understanding not just of the faults that are produced by a classifier but also of the errors that are being made.

```
In [32]: plt.figure(figsize = (10,10))  
sns.heatmap(cm,cmap= "Blues", linecolor = 'black' , linewidth = 1 , annot = True, fmt='',xticklabels = labels,yticklabels = labels)
```

Out[32]: <AxesSubplot:>



10 Project Management

In order to finish on time and provide a conclusion, there are many duties that need to be completed. Once a week, we should review the progress that has been made on the project. During the course of the study, we will be officially required to reveal two deliverables, including a request for approval of our research ethics on the 11th of October and our thesis on the 9th of December. Everything in the allocated time for this assignment started at the very beginning of the semester and continued all the way to its conclusion. The main research plan includes the activities that are explained further below. In addition, the Gantt Chart for this project is shown below.

Project Schedule

Task 1- Gathering and Examination of Data

Task 2- Pre-processing of Data

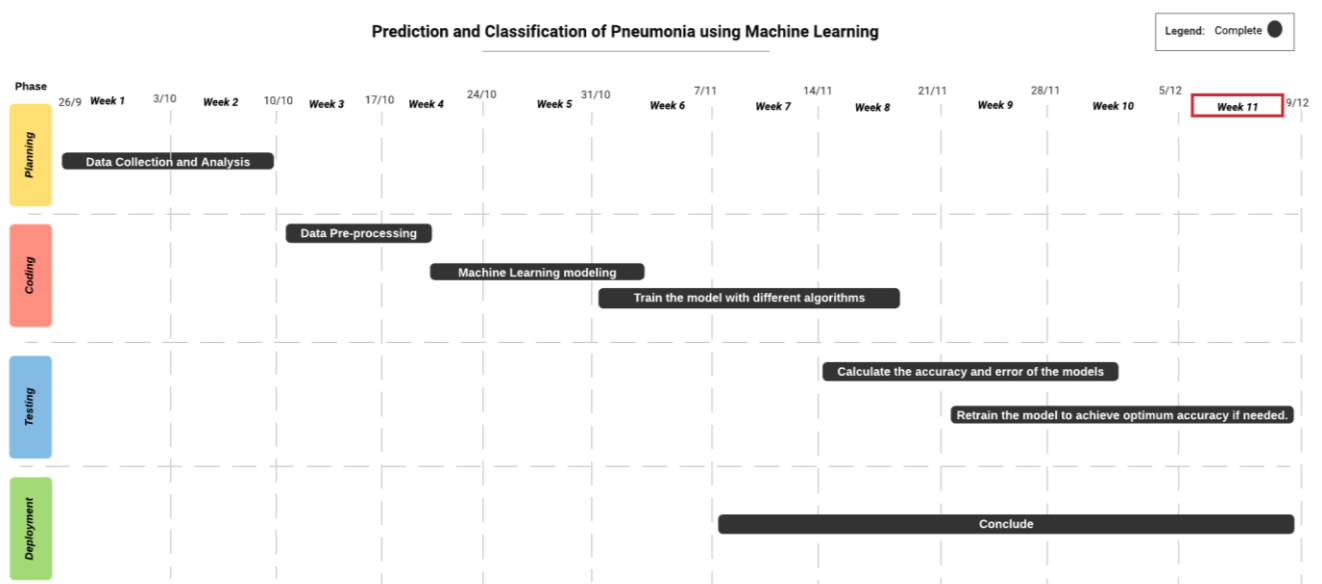
Task 3- Modelling of Machine Learning Algorithms

Task 4- Train the model with various algorithms

Task 5- Evaluate the accuracy and error of the models

Task 6- Retrain the model to achieve optimum accuracy if required.

Task 7- Documentation and Conclusion



Risk Management

The programme never collected or stored any private information about individuals, with the exception of data that was made publicly available. Time management was the one and only concern that occurred throughout the course of the project; nevertheless, with the guidance and assistance provided by the supervisor, this problem was successfully handled and managed.

Quality Management

Python was utilised all across entire existence of the development process for the side. Since Python is the most effective choice for resolving AI problems, it is reasonable to conclude that those of the greatest quality have chosen to use it. The next significant step is to choose libraries that are suitable for use with such an approach. TensorFlow and keras are open-source software packages created by Google to solve problems in artificial intelligence and machine learning. The core algorithms are brought in from outside sources and then implemented using the Keras library. Because of Google's importance, the quality of the libraries that were chosen is of the best possible standards.

As far as the platform and the tools are concerned, the Python application is completely programmed and then run in Jupyter Notebook. By default, JupyterLab (also known as AI Notebook) on GCP is configured to use no more than 3.2 gigabytes of RAM. The Jupyter Kernel will be said to "die" if a significant quantity of data is put into memory (that is, by RAM).

Memory and storage space requirements are as follows, per user: 512 MB RAM, 1 GB disc space plus. 5 CPU core.

Server overhead is between 2 and 4 gigabytes, or 10 percent of the total system overhead, which ever is bigger. 5 CPU cores, 10GB storage space.

To make it easier to grasp the code lines, the most important phases have been explained using comments.

Social, Legal, Ethical and Professional Considerations

In order for the University to consider the idea, one must first get ethical clearance. A questionnaire was filled out to address this issue since it was required as part of the approval procedure. Immediately after the completion and submission of the questionnaire, this was subjected to vetting and evaluation by authorised individuals such as the Project supervisor and the Module leader. It was investigated despite the fact that the research revealed no particularly serious risk. As a direct consequence of this, the ethical standards of the project are in agreement with the regulations of the institution. Aside from that, the dataset was the most important source of data for the research since it was the only one that featured photos of lungs taken from the web source. As a direct consequence of this, there will be no ethical, social, or professional problems raised either now or in the foreseeable future.

11 Critical Appraisal

It may be a challenging endeavour to effectively implement computer vision utilising neural networks or machine learning Learning architecture. This project's objective was to identify and implement an improved machine learning model that could predict and classify pneumonia symptoms in real time. Producing such an image collection and applying suitable preprocessing methods to it within the time restriction that was allotted for the project was a challenging endeavour in and of itself. As a direct consequence of this, the results of these efforts exceeded all expectations.

In terms of overall model correctness, the TensorFlow Keras algorithm achieved an accuracy of 93% respectively. These are good image detection model accuracies. This was attainable due to the high quality of the pre-trained models as well as the substantial amount of photos included inside the dataset.

Overall, the project is developed to the best of its abilities despite the time limits and other considerations that were taken into account.

12 Conclusions

A very accurate strategy for predicting the recurrence of pneumonia has been discovered as a result of extensive research, the diligent study of a large number of papers, and comprehensive understanding of customer concerns. An exceptionally accurate prediction of pneumonia has been developed with the use of the Keras (93%). According to research on the predicting and classification of pneumonia, there is not yet a quality factor system in place. Though many methods have been developed to work on this dataset, the proposed methodology achieved better results.

The results of this research may lead us to the conclusion that the models that were employed here may be deployed to an image dataset in order to enhance the accuracy and minimise the amount of time required for processing.

Achievements

After conducting extensive research, reviewing a number of articles, and taking into account the challenges faced by patients, a method that has proven to be very effective in predicting and classifying cases of pneumonia has been developed. With the help of the Keras Tensor Flow, a very accurate prediction of pneumonia could now be made. According to the research done on pneumonia prediction, there is not an adequate warning system that is currently in place. According to the findings of this study, the models that were employed here have the potential to be applied to an image dataset in order to enhance the accuracy and speed up processing.

Future Work

It would be fascinating to see approaches in the future that enable the weights corresponding to various models to be estimated more efficiently, as well as a model that takes into account of individual patient's history when generating predictions. This might be accomplished in the future.

It is possible to create mobile apps, including with other labels in furthermore to normal and pneumonia. In addition to this, it is essential to have a system that can adapt to the many characteristics of the data set while still executing with a high level of accuracy.

13 Student Reflections

This work was different from anything I had ever done before in terms of scale and complexity, but it was a fantastic opportunity to boost my knowledge and skills, all of which are unquestionably transferrable and helpful in my future studies and profession. The work that I did was to expand my knowledge and skills was a fantastic opportunity. In addition, I am certain that I was able to successfully manage the project in a well-organized way, despite the fact that the time constraint required me to make a few minor adjustments to the plan that I had developed. The reason for this is that I made an effort to stick to my management plan while also keeping in touch with my supervisor. This provided me with the opportunity to gain weekly assistance and benefits from their extensive experience.

During the time that I was working on the project, I not only examined it but also did more research on it. While I was working on it, I came up with some great ideas for the future works of all of the aforementioned. The part of the article that was about future work was helpful to me in broadening my knowledge of other models, too. Another noteworthy accomplishment is the growth of computer vision and the accompanying expansion of possible solutions. The TensorFlow object identification API provided me with a wealth of inspiration for additional projects that I could undertake in my spare time as a hobby.

Bibliography and References

1. *AttributeError: Module 'TensorFlow' has no attribute '__version__'*. (n.d.). Stack Overflow. <https://stackoverflow.com/questions/61554548/attributeerror-module-tensorflow-has-no-attribute-version>
2. *Installing Python and TensorFlow with Jupyter notebook configurations*. (2022, March 5). Python-bloggers. <https://python-bloggers.com/2022/03/installing-python-and-tensorflow-with-jupyter-notebook-configurations/>
3. *Jupyter notebooks requirements*. (n.d.). TIBCO Product Documentation. https://docs.tibco.com/pub/sfire-dsc/6.5.0/doc/html/TIB_sfire-dsc_sys-req/GUID-291ABBD3-9DC6-4659-8595-3F208F24565A.html
4. Keras Team. (n.d.). *Keras documentation: VGG16 and VGG19*. Keras: the Python deep learning API. <https://keras.io/api/applications/vgg/>
5. Mac, K. (2021, September 8). *Increasing memory in Google cloud platform AI notebook JupyterLab settings*. Data Analytics. [https://dsstream.com/increasing-memory-in-google-cloud-platform-ai-notebook-jupyterlab-settings/#:~:text=JupyterLab%20\(or%20AI%20Notebook\)%20in,Jupyter%20Kernel%20will%20%E2%80%9Cdie%E2%80%9D](https://dsstream.com/increasing-memory-in-google-cloud-platform-ai-notebook-jupyterlab-settings/#:~:text=JupyterLab%20(or%20AI%20Notebook)%20in,Jupyter%20Kernel%20will%20%E2%80%9Cdie%E2%80%9D)
6. *Project Jupyter*. (2022, November 19). Wikipedia, the free encyclopedia. Retrieved December 9, 2022, from https://en.wikipedia.org/wiki/Project_Jupyter
7. *Project Jupyter*. (2022, November 19). Wikipedia, the free encyclopedia. Retrieved December 9, 2022, from https://en.wikipedia.org/wiki/Project_Jupyter
8. (n.d.). Project Jupyter. <https://jupyter.org/>
9. (n.d.). TensorFlow. <https://www.tensorflow.org/>
10. Thakur, R. (2020, November 24). *Step by step VGG16 implementation in Keras for beginners*. Medium. <https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c>
11. Yegulalp, S. (n.d.). *What is TensorFlow? The machine learning library explained*. InfoWorld. <https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html>
12. *CNN-based deep learning model for chest X-ray health classification using TensorFlow*. (n.d.). IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/9140733>

13. *Diagnosis of pneumonia from chest X-ray images using deep learning.* (n.d.). IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/8741582>
14. *Efficient pneumonia detection in chest xray images using deep transfer learning.* (2020, June 19). MDPI. <https://www.mdpi.com/2075-4418/10/6/417>
15. *Feature extraction and classification of chest X-ray images using CNN to detect pneumonia.* (n.d.). IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/9057809>
16. *Just a moment...* (n.d.). Just a moment... <https://www.sciencedirect.com/science/article/pii/S0263224120305844>
17. *Pneumonia detection in chest X-ray images using an ensemble of deep learning models.* (2021, September 7). PLOS. <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0256630>
18. *Why use Python for AI and machine learning?* (2022, August 31). SteelKiwi | Web and Mobile Software Development Company. <https://steelkiwi.com/blog/python-for-ai-and-machine-learning/>
19. *Intelligent diagramming.* (n.d.). Lucid-chart. https://www.lucidchart.com/pages/landing?utm_source=google&utm_medium=cp c&utm_campaign= chart en tier1 mixed search brand exact &km CPC CampaignI d=1490375427&km CPC AdGroupID=55688909257&km CPC Keyword=lucid%20c hart&km CPC MatchType=e&km CPC ExtensionID=&km CPC Network=g&km C PC AdPosition=&km CPC Creative=442433236001&km CPC TargetID=aud- 1434328328387:kwd- 55720648523&km CPC Country=9046172&km CPC Device=c&km CPC placemen t=&km CPC target=&gclid=CjwKCAiAs8acBhA1EiwAgRFdw9h_rWQaRwpzmQ4nu kmGKQHRk4hIb0hVCk086TFR_5wxFiXaly-amBoCy5QQA vD_BwE
20. *It is always necessary to include a flatten layer after a set of 2D convolutional layers for convolutional neural networks in Keras?* (n.d.). Cross Validat- ed. <https://stats.stackexchange.com/questions/506796/it-is-always-necessary-to-include-a-flatten-layer-after-a-set-of-2d-convolutiona>
21. *Keras flatten.* (2022, June 4). EDUCBA. <https://www.educba.com/keras-flatten/>
22. *Keras conv2D.* (2022, April 14). EDUCBA. <https://www.educba.com/keras-conv2d/>
23. Rosebrock, A. (2021, April 17). *Keras Conv2D and Convolutional layers.* PyImageSearch. <https://pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/>
24. *Keras - Dense layer.* (n.d.). Online Tutorials Li- brary. https://www.tutorialspoint.com/keras/keras_dense_layer.htm

25. *Keras dense layer explained for beginners.* (2020, October 20). *MLK - Machine Learning Knowledge*. <https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/>
26. *Keras MaxPooling2D.* (2022, October 4). *EDUCBA*. <https://www.educba.com/keras-maxpooling2d/>
27. KUMAR, B. (2021, February 10). *Image Preprocessing - Why is it necessary?* *Medium*. <https://medium.com/spidernitt/image-preprocessing-why-is-it-necessary-8895b8b08c1d>
28. *Confusion matrix in machine learning with example.* (2022, November 19). *Guru99*. <https://www.guru99.com/confusion-matrix-machine-learning-example.html>
29. *Confusion matrix in machine learning.* (n.d.). *www.javatpoint.com*. <https://www.javatpoint.com/confusion-matrix-in-machine-learning>
30. *Horizontal and vertical flip data augmentation.* (2021, April 6). *Studytonight - Best place to Learn Coding Online*. <https://www.studytonight.com/post/horizontal-and-vertical-flip-data-augmentation>
31. *Just a moment...* (n.d.). *Just a moment...* <https://www.sciencedirect.com/topics/engineering/confusion-matrix>
32. Brownlee, J. (2022, August 15). *Difference between a batch and an epoch in a neural network.* *Machine Learning Mastery*. <https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>
33. SHARMA, S. (2019, March 5). *Epoch vs batch size vs iterations.* *Medium*. <https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9>
34. Chase, C. (n.d.). *Bacterial pneumonia: Symptoms, causes, and treatment.* *Healthline*. <https://www.healthline.com/health/bacterial-pneumonia#prevention>

Appendix A – Interim Progress Report and Meeting Records

Meeting No.	Discussed Topics	Date	Time
1	1.Project idea discussion 2.Data set discussion	27-Sep-2022	15:00-15:07
2	1.Data set approval 2.Project title finalized 3.Ethics approval	18-Oct-2022	15:00-15:10
3	1. Meeting Log 2.Aim & objective discussion 3.Introduction for Project 4.Discussion about TensorFlow method 5. Discussion about Kares library 6. To Do -->Literature review, Methodology, Flowchart	25-Oct-2022	15:00-15:10
4	1.Update Meeting Log 2.Literature review, Methodology, Flowchart 3.Flowchart discussion 4. Report writing	08-Nov-2022	15:00-15:15
5	1. Updated Meeting Log 2. Discussion on Literature review, Methodology, Flowchart 3. Initialization of the coding part 4. Discussion about report writing	22-Nov-2022	15:00-15:20

Appendix B – Certificate of Ethics Approval

Prediction and Classification of Pneumonia using Machine learning

P143068



Certificate of Ethical Approval

Applicant: Harni Ramakrishnan
Project Title: Prediction and Classification of Pneumonia using Machine learning

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval: 14 Oct 2022
Project Reference Number: P143068

Appendix C – Dataset and Code

The link to the dataset and the python code is:

Dataset: <https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia?datasetId=17810>

Code: https://gitlab.com/Harni_Ramakrishnan/12141340_dissertation