

**An Intelligent System for Covid19 Detection Through Chest X-ray Images Using Deep Learning Algorithms**

*A project report submitted in partial fulfilment of the requirement for the award of degree of*

**BACHELOR OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

*Submitted*

*by*

**A.B.S.L.REKHA (17341A0506)**

**Ch.PARDHA SARADHI (17341A0536)**

**Ch.NAVEEN (17341A0535)**

**B.SRAVYA VANDANA (17341A0514)**

**G.CHINMAI (17341A0553)**

*Under the esteemed guidance of*

**Mr.Panigrahi Srikanth**

Assistant Professor

Dept. of CSE

**GMR Institute of Technology**

**An Autonomous Institute Affiliated to JNTUK, Kakinada**

(Accredited by NBA, NAAC with ‘A’ Grade & ISO 9001:2008 Certified Institution)

**GMR Nagar, Rajam – 532127,**

**Andhra Pradesh, India**

**July 2021**



**Department of Computer Science and Engineering**

**CERTIFICATE**

This is to certify that the thesis entitled **An Intelligent System for Covid-19 Detection Through Chest X-ray Images Using Deep Learning Algorithms** submitted by **A.B.S.L.Rekha(17341A0506), Ch.Pardha Saradhi (17341A0536), Ch.Naveen(17341A0535), B.Sravya Vandana(17341A0514), G.Chinmai(17341A0553)** has been carried out in partial fulfilment of the requirement for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **GMRIT, Rajam** affiliated to **JNTU KAKINADA** is a record of bonafide work carried out by them under my guidance & supervision. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree.

**Signature of Supervisor Signature of HOD**

**Mr. Panigrahi Srikanth Dr. A. V. Ramana**

Assistant Professor Professor & Head

Department of CSE Department of CSE

GMRIT, Rajam. GMRIT, Rajam.

The report is submitted for the viva-voce examination held on ………………..

Signature of Internal Examiner Signature of External Examiner

**ACKNOWLEDGEMENT**

It gives us an immense pleasure to express deep sense of gratitude to my guide **Mr. Panigrahi Srikanth,** Assistant Professor, Department of Computer Science and Engineering for his whole hearted and invaluable guidance throughout the project work. Without his sustained and sincere effort, this project work would not have taken this shape. He encouraged and helped us to overcome various difficulties that we have faced at various stages of our project work.

We would like to sincerely thank our Head of the department **Dr. A. V. Ramana**, for providing all the necessary facilities that led to the successful completion of our project work.

**We would like to take this opportunity to thank our beloved Vice-Principal Dr. J.Raja Murugadoss, for providing a great support to us in completing our project and for giving us the opportunity of doing the project work.**

We would like to take this opportunity to thank our beloved Principal **Dr.C.L.V.R.S.V.Prasad**, for providing all the necessary facilities and a great support to us in completing the project work.

We would like to thank all the faculty members and the non-teaching staff of the Department of Computer Science and Engineering for their direct or indirect support for helping us in completion of this project work.

Finally, we would like to thank all of our friends and family members for their continuous help and encouragement.

**A.B.S.L.Rekha (17341A0506)**

**Ch. Pardha Saradhi (17341A0536)**

**Ch.Naveen (17341A0535)**

**B.Sravya Vandana (17341A0514)**

**G.Chinmai (17341A0553)**

**ABSTRACT**

COVID-19 is a disease that causes respiratory issues and causes deaths around the world. Studies are ongoing for the diagnosis and treatment of this disease, which is defined as a pandemic. The expeditious spread of Covid19 disease has become an example of the worst disruptive disasters of the century around the globe. It first appeared in china. Clinical image analysis of chest X-Ray images can play an important role for a precise diagnostic using deep learning algorithms. Specifically, chest X-Ray images can be analyzed to identify the presence of COVID-19 in a patient. In this work, the deep learning models were used, including Conventional Neural networks(CNN), Residual Networks(ResNet), and **Anamorphic Depth Embedding-Based Lightweight CNN(AnamNet).** The results were compared after conducting several experiments. Based on results, the model is selected for application. An application was developed that is integrated to the model which can be used to identify the covid 19 using chest X-ray images and also improving the system performance.

**Keywords:** Deep learning algorithm, Covid19, Chest X-Ray Images and Diagnosis.

**TABLE OF CONTENTS**

**ACKNOWLEDGEMENT** 3

**ABSTRACT** 4

**TABLE OF CONTENTS**

**LIST OF TABLES**

**LIST OF FIGURES** 7

**LIST OF SYMBOLS & ABBREVIATIONS** 9

**1. INTRODUCTION** 10

1.1 DEEP LEARNING 10

1.2 MACHINE LEARNING

1.3 TRANSFER LEARNING

1.4 ENSEMBLE LEARNING

1.5 STATISTICS

1.6 WEB DEVELOPMENT

1.7 CHAT BOT

**2. LITERATURE SURVEY** 14

**3. METHODOLOGY** 17

3.1 PROPOSED WORK

3.2 DATA AUGMENTATION

3.3 CNN

3.4 RESNET

3.5 VGG-19

3.6 INCEPTION NET

3.7 INCEPTION RESNET

3.8 ENSEMBLE LEARNING

3.9 HTML

3.10 CSS

3.11 BOOTSTRAP

3.12 DJANGO

3.13 IBM CLOUD

3.13.1 IBM WATSON

**4.IMPLEMENTATION**

4.1 DATASET

4.2 TOOLS AND MODULES

4.3 SOURCE CODE

4.3.1 DATA SPLITTING

4. 3.2 DATA AUGMENTATION

4.3.3 CNN

4.3.4 RESNET50

4.3.5 VGG-19

4.3.6 MODEL CHECKPOINTS

4.3.7 MODEL TRAINING

4.3.8 PERFORMANCE ANALYSIS

4.3.9 ENSEMBLE WITH VOTING

**5. RESULTS**

5.1 PERFORMANCE OF MODELS

5.2 OUTPUT

**6. CONCLUSION AND FUTURE WORK**

**7.REFERENCES** 63

**LIST OF TABLES**

**Table No. Table Name Page**

5.1.1 Performance of deep learning models

**LIST OF FIGURES**

**FIGURE NO TITLE PAGE NO**

1.4.1 Countries Cases Distribution

1.4.2 Coronavirus Daily New Cases

1.4.3 Coronavirus Daily Deaths

3.1.1 Proposed system Architecture for Covid-19 detection

3.3.1 Basic CNN Architecture

3.4.1 ResNet Architecture

3.4.2 Residual Block

3.5.1 VGG-19 Architecture

3.6.1 Inception Block-A

3.6.2 Inception Block-B

3.6.3 Inception Block-C

3.6.4 Inception with Auxiliary Classifier

3.6.5 Inception-v3 Architecture

3.7.1 Inception ResNet Architecture

3.7.2 Residual Inception Blocks

3.7.3 Scaling of Residuals

3.8.1 Ensemble Blocks

3.13.1.1 IBM Watson

4.0.1 Proposed Ensemble architecture

4.1.1 Covid-19 x-ray images

4.1.2 Normal x-ray images

4.3.2.1 Augmented Images

4.3.3.1 Custom CNN

5.1.2 Performance Analysis of Individual Models

5.1.3 Comparison of Accuracy obtained from trained models

5.1.4 Comparison of Recall obtained from trained models

5.1.5 Comparison of Precision obtained from trained models

5.1.6 Comparison of F1-score obtained from trained models

**LIST OF SYMBOLS & ABBREVIATIONS**

SARS-CoV-2 : Severe Acute Respiratory Syndrome CoronaVirus2

WHO : World Health Organization

COVID-19 : CoronaVirus Disease-2019

CNN : Convolutional Neural Networks

ReLU : Rectified Linear Units

RESNET : Residual Neural Networks

VGG-19 : Visual Geometry Group-19

HTML : Hyper Text Markup Language

CSS : Cascading Style Sheet

PHP : Hypertext Preprocessor

IDE : Integrated Development Environment

IBM : International Business Machines

**CHAPTER-1**

**1.INTRODUCTION**

In December 2019, a series of pneumonia cases of unknown cause appeared in Wuhan (Hubei, China). After a few weeks, in January 2020, deep sequencing analysis from lower respiratory tract samples identified a novel virus severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) as causative agent for that observed pneumonia cluster. On February 11th, 2020, the World Health Organization (WHO) Director-General, Dr. Tedros Adhanom Ghebreyesus, named the disease caused by the SARS-CoV-2 as COVID-19, and by March 11th, 2020 when the number of countries involved was 114, with more than 118,000 cases and over 4000 deaths, the WHO declared the pandemic status. Due to high virus infection, it is essential to control the disease, including rapid diagnosis and timely quarantine. Currently, reverse transcription-polymerase chain reaction (RT-PCR) is the standard test for diagnosing COVID-19. But with less availability of RT-PCR kit, it is better to find using chest x-ray images. Convolutional neural networks(CNN) and advanced CNN are used to identify the abnormalities present in a lung.

* 1. **Deep Learning**

Deep learning is a subcategory of machine learning focused on parameterizing multilayer neural networks that can learn representations of the data with multiple layers of abstractions.

It is completely based on artificial neural networks, as neural networks are going to mimic the human brain so deep learning is also a kind of mimic of the human brain. In deep learning, we don’t need to explicitly program everything. In the human brain approximately 100 billion neurons all together this is a picture of an individual neuron and each neuron is connected through thousands of their neighbors. The question here is how do we recreate these neurons in a computer. So, we create an artificial structure called an artificial neural net where we have nodes or neurons. We have some neurons for input value and some for-output value and in between, there may be lots of neurons interconnected in the hidden layer.

The Deep Neural Network is more creative and complicated than the neural network. Deep Neural Network algorithms can recognize sounds and voice commands, make predictions, think creatively, and do analysis. One of the main advantages of deep learning lies in being able to solve complex problems that require discovering hidden patterns in the data and/or a deep understanding of intricate relationships between a large number of interdependent variables.

* 1. **Machine Learning**

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Types of machine learning problems:

**1.2.1 Supervised learning**

The computer is presented with example inputs and their desired outputs, given by a “teacher”, and the goal is to learn a general rule that maps inputs to outputs. The training process continues until the model achieves the desired level of accuracy on the training data. Some real-life examples are:

Image Classification: You train with images/labels. Then in the future you give a new image expecting that the computer will recognize the new object.

Market Prediction/Regression: You train the computer with historical market data and ask the computer to predict the new price in the future.

**1.2.2 Unsupervised learning**

No labels are given to the learning algorithm, leaving it on its own to find structure in its input. It is used for clustering populations in different groups. Unsupervised learning can be a goal in itself (discovering hidden patterns in data).

Clustering: You ask the computer to separate similar data into clusters, this is essential in research and science.

High Dimension Visualization: Use the computer to help us visualize high dimensional data.

**1.2.3 Semi-supervised learning**

Problems where you have a large amount of input data and only some of the data is labeled, are called semi-supervised learning problems. These problems sit in between both supervised and unsupervised learning. For example, a photo archive where only some of the images are labeled, (e.g. dog, cat, person) and the majority are unlabeled.

**1.2.4 Reinforcement learning**

A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

**1.3 Transfer Learning**

Transfer learning is the reuse of a pre-trained model on a new problem. It's currently very popular in [deep learning](https://builtin.com/artificial-intelligence/deep-learning) because it can train deep [neural networks](https://builtin.com/data-science/recurrent-neural-networks-and-lstm) with comparatively little data. This is very useful in the [data science](https://builtin.com/data-science) field since most real-world problems typically do not have millions of labeled data points to train such complex models. Transfer learning is mostly used in computer vision and natural language processing tasks like sentiment analysis due to the huge amount of computational power required.Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks, and not needing a lot of data.

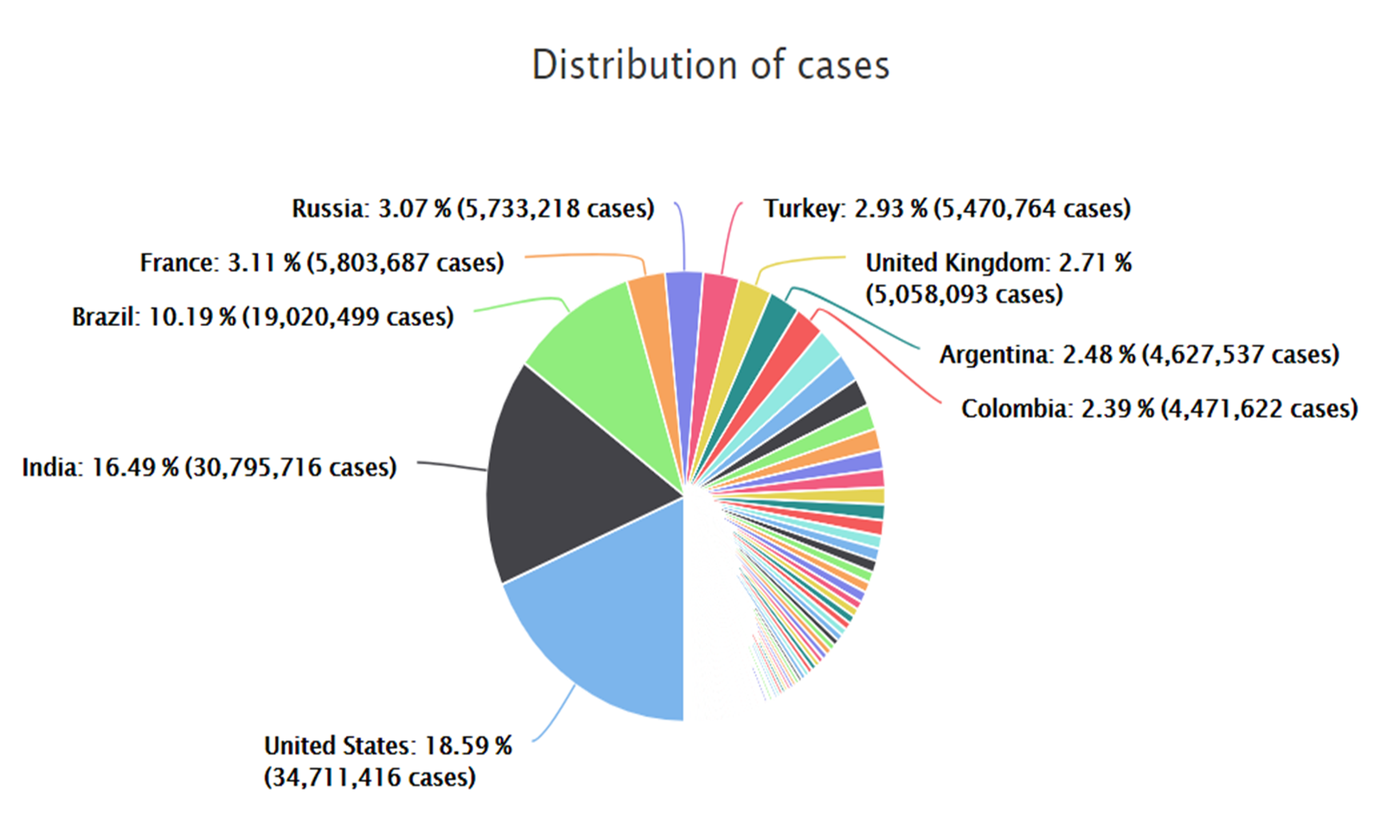
**1.4 Ensemble Learning**

Ensemble means a group of elements viewed as a whole rather than individually. Ensemble learning is a form of hybrid learning system in which multiple analytics are combined intelligently with the purpose of obtaining better (more accurate, more robust, etc.) results than a single analytics can provide. Ensemble learning helps improve machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model.

**1.5 Statistics**

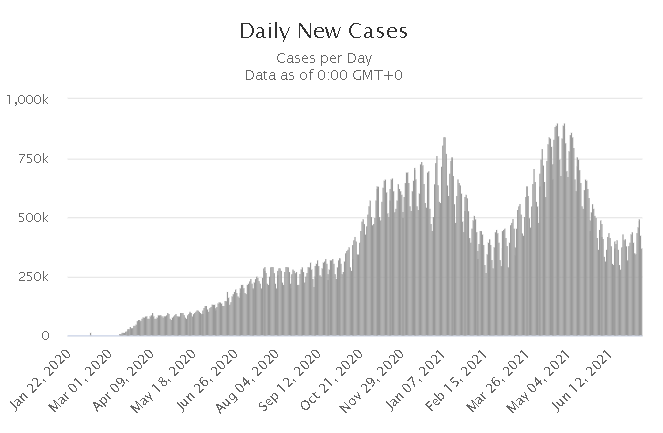
The following graphs represent the number of new cases, number of deaths per day in the whole world and number of new cases occurring in each country.

**1.5.1 Countries Cases Distribution**

****

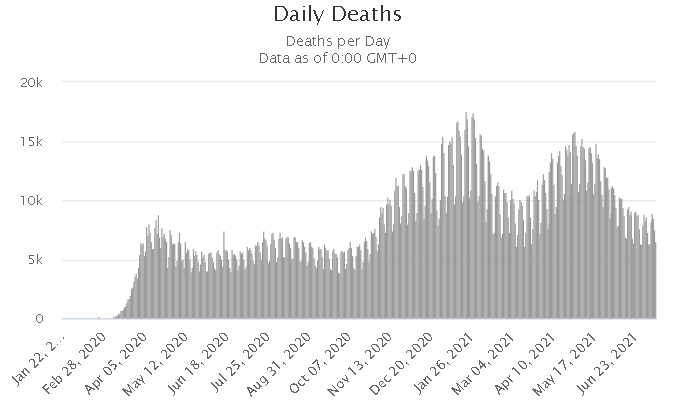
**Fig-1.4.1 Countries Cases Distribution**

**1.5.2 Coronavirus Daily New Cases**

****

**Fig-1.4.2 Coronavirus Daily New Cases**

**1.5.3 Coronavirus Daily Deaths**

****

**Fig-1.4.3 Coronavirus Daily Deaths**

**1.6 Web Development**

Web development is the process of building and maintaining a [website](https://www.webopedia.com/definitions/web-site/) for the [Internet](https://www.webopedia.com/definitions/internet/) or an [intranet](https://www.webopedia.com/reference/i/Intranet.html) that is accessed through a [web browser](https://www.webopedia.com/definitions/browser/) and [hosted](https://www.webopedia.com/definitions/hosted/) on a [server](https://www.webopedia.com/definitions/server/), either on on-premises [hardware](https://www.webopedia.com/definitions/hardware/) or in the [cloud](https://www.webopedia.com/definitions/cloud/).

This includes everything from single [plain-text](https://www.webopedia.com/definitions/plain-text/) [webpages](https://www.webopedia.com/definitions/web-page/) up to complex [web applications](https://www.webopedia.com/definitions/application/).Web developers build websites by coding using different [programming languages](https://www.webopedia.com/definitions/programming-language/). These languages consist of unique vocabularies, syntax and [commands](https://www.webopedia.com/definitions/command/) that define the visual representation and functionality of websites.

Some of the most common coding languages include the following:

* [HTML](https://www.webopedia.com/definitions/html/)
* CSS
* [JavaScript](https://www.webopedia.com/definitions/javascript/)
* Python
* [C](https://www.webopedia.com/definitions/c-language/)/[C++](https://www.webopedia.com/definitions/c-plus-plus/)
* [C#](https://www.webopedia.com/definitions/c-sharp/)

### Front-end

Front-end, also known as [client-side](https://www.webopedia.com/definitions/client-side/), development is used to build the [layout](https://www.webopedia.com/definitions/layout/), design and [interactivity](https://www.webopedia.com/definitions/interactive/) of a website. Front-end development is used to define how a website displays videos, images, text and [graphics](https://www.webopedia.com/definitions/graphics/). It also defines front-facing interaction, such as minimizing and maximizing visual assets, highlighting text and filling out form fields. Front-end development uses the programming languages HTML, CSS and Javascript.

### Back-end

Back-end, or [server-side](https://www.webopedia.com/definitions/server-side/), development builds the digital [infrastructure](https://www.webopedia.com/definitions/infrastructure/) and behind-the-scenes functionality of a website to ensure it runs smoothly. The back end is made up of the server the website is hosted on, an application that operates the site and a [database](https://www.webopedia.com/definitions/database/) that stores the site [data](https://www.webopedia.com/definitions/data/). Developers can use a variety of programming languages for back-end development, as servers can be configured to understand virtually any language.

**1.7 CHAT BOT**

A Conversational Bot or Chatbot is a virtual assistant that acts as an intelligent intermediary between people, digital systems, and internet-enabled things. It replaces the traditional Graphical User Interfaces (GUIs) of an application or website with a Conversational User Interface. It is a paradigm shift from the earlier communications achieved either by entering syntax-specific commands or clicking icons.

Chatbots are designed to chat with users through a combination of natural language-based conversations. Responses come in the form of buttons, calendars, or other widgets that accelerate the speed with which a user can respond.

**CHAPTER-2**

**2. LITERATURE SURVEY**

In the literature,this author Naveen Paluru(2021) et.al[1],proposed anamorphic depth embedding-based lightweight CNN, called Anam-Net, to segment anomalies in COVID-19 chest CT images. And they concluded that the Anam-Net was benchmarked against other state-of-the-art lightweight and heavy networks, such as ENet, UNet++, SegNet, Attention UNet, LEDNet, and DeepLabV3+. The proposed Anam-Net was also deployed on embedded systems, such as Raspberry Pi 4, NVIDIA Jetson Xavier, and mobile-based Android application (CovSeg) embedded with Anam-Net to demonstrate its suitability for point-of-care platforms.

In this author**Murat Canayaz(2021) et.al[2]**, has proposed a deep learning-based approach that can be used for early diagnosis of the disease. Pre-processing was performed using the image contrast enhancement algorithm on the prepared data set and a new data set was obtained. Feature extraction was completed from this data set with deep learning models such as AlexNet, VGG19, GoogleNet, and ResNet. For the selection of the best potential features, two metaheuristic algorithms of binary particle swarm optimization and binary gray wolf optimization were used. The results proved that the approach they proposed can help experts during Covid-19 diagnostic studies.

In **Ezzat,D.,Hassanien(2020) et.al[3]** has provided an approach called GSA-DenseNet121-COVID-19 that can be used to diagnose COVID-19 cases through chest X-ray images. The proposed GSA-DenseNet121-COVID-19 consists of four main stages: the data preparation stage, the hyperparameters selection stage, the learning stage, and the performance measurement stage. The proposed approach was compared to more than one approach, and the results of the comparison showed the effectiveness of the proposed approach in diagnosing the COVID-19.

**Gómez-Ríos(2020) et.al[4]**, In this paper introduced a dataset, named COVIDGR-1.0, with high clinical value. COVIDGR-1.0 includes the four main COVID severity levels identified by a recent radiological study. They proposed a methodology, called COVID-SDNet, that combines segmentation, data-augmentation and data transformation. The obtained results show the high generalization capacity of COVID-SDNet, especially on severe and moderate levels as they include important visual features.

**Wu, Y. H (2021) et.al[5]**, this author to facilitate the training of strong CNN models for COVID19 diagnosis, in this paper, they systematically constructed a large scale COVID-19 Classification and Segmentation (COVID-CS) dataset. They developed a Joint Classification and Segmentation (JCS) system for COVID-19 diagnosis. In this system, the classification model identified whether the suspected patient is COVID-19 positive or not, along with convincing visual explanations.

**Bhawna Nigama(2021) et.al[6]**, in this paper, popular and best performing deep learning architectures are used for COVID-19 detection in the suspected patients by analyzing the X-ray images. Radiologic images such as X-ray or CT scans consist of vital information. The models performed efficiently and provided considerable results to many SoTA Coronavirus detection systems. In this work, various SoTA deep learning architectures are used to perform COVID-19 chest X-rays. The highest recognition accuracy is achieved from the EfficientNet model, i.e., 93.48%. It is observed that deep learning models provide better and faster results by analysing the image data to identify the presence of COVID in a person.

**Boran Sekeroglu (2020) et.al[7]** in this study, several experiments were performed for the high-accuracy detection of COVID-19 in chest X-ray images using ConvNets. Various groups COVID-19/Normal, COVID-19/Pneumonia, and COVID-19/Pneumonia/Normal were considered for the classification. The results showed that the convolutional neural network with minimized convolutional and fully connected layers is capable of detecting COVID-19 images within the two-class, COVID-19/Normal and COVID-19/Pneumonia classifications, with mean ROC AUC scores of 96.51 and 96.33%, respectively.

**Rachna Jain (2020) et.al[8]**, in this work, they experimented with multiple CNN models in an attempt to classify the Covid-19 affected patients using their chest X-ray scans. We have compared Inception V3, Xception, and ResNeXt models and examined their accuracy.Further, they concluded that out of these three models, the XCeption net has the best performance and is suited to be used.

**Worapan Kusakunniran (2021) et.al[9]**, this paper presents a solution for COVID-19 classification in chest x-ray images. Its backbone CNN architecture is developed using ResNet-101. The model is trained from scratch with a large size of the network’s input of 1500 × 1500 pixels. Data augmentation is also applied on the original training images to enhance the regularization of the model. The proposed solution achieves very promising sensitivity, specificity, and accuracy of 97%, 98%, and 98%, respectively. The developed solution can also generate the heatmap with a confidence score of being COVID-19, to emphasize the result on each test image. The heatmap is visualized on only lung regions segmented using U-Net.

**Ahmed Hamza OsmanID (2021) et.al[10]**, in this study, the SOM-LWL model is suggested for diagnosis and detection of the COVID-19 disease based on chest X-rays. The proposed SOM-LWL model improved the correlation coefficient performance results between the Covid19, no-finding, and pneumonia cases; pneumonia and no-finding cases; Covid19 and pneumonia cases; and Covid19 and no-finding cases from 0.9613 to 0.9788, 0.6113 to 1 0.8783 to 0.9999, and 0.8894 to 1, respectively. The proposed LWL-SOM had better results for discriminating COVID-19 and non-COVID-19 patients than the current machine learning-based solutions using AI evaluation measures.

**Nikhil Bhatia,(2021) et.al[11]**, in this study, the performance of several pretrained architectures were reviewed for detecting the radiographic features in X-rays. After analyzing the pre-trained architectures, Squeezenet1.1, AlexNet, DenseNet-121, GoogleNet were better models than the others in identifying the coronavirus in the chest X-rays.

[**Chaimae Ouchicha**](https://www.sciencedirect.com/science/article/abs/pii/S096007792030641X#!)**, 2020) et.al[12]**, in this paper, they proposed a novel deep convolutional neural network (CVDNet) model for the detection of COVID-19 cases, in order to distinguish more precisely the patients affected by COVID19 from healthy persons and viral pneumonia patients using chest X-ray images. It was observed that CVDNet achieved an average accuracy of 97.20% for detecting COVID-19 and an average accuracy of 96.69% for three-class classification (COVID-19 vs. normal vs. viral pneumonia), which exhibit superior and promising performance in classifying COVID-19 cases.

**Soumya Ranjan Nayak (2021) et.al[13]**, in this study, a DL based automated method was proposed for efficient classification of COVID-19 infection cases from normal cases using chest X-ray images. The results indicated that ResNet-34 outperformed other competitive networks with an accuracy of 98.33% and can hence be regarded as a potential model for prediction of COVID-19 infection.

**Elene Firmeza Ohata (2020) et.al[14]**, in this work, they proposed an automatic detection method for COVID-19 infection based on chest X-ray images. The datasets constructed for this study are composed of 194 X-ray images of patients diagnosed with coronavirus and 194 X-ray images of healthy patients. And the results showed that the use of CNNs to extract features, applying the transfer learning concept, and then classifying these features with consolidated machine learning methods is an effective way to classify X Ray images as in normal conditions or positive for COVID-19.

[**Khandaker Foysal Haque**](https://ieeexplore.ieee.org/author/37086848753) **(2020) et.al[15]**, a CNN based model is proposed in this paper for detecting COVID-19 cases from patients chest X-rays.This model performs with accuracy and precision of 97.56% and 95.34% respectively.Moreover, this model is compared to the other two CNN models with a different number of convolutional layers. The comparative studies show better F1-score and overall performance of the proposed model than that of the other two.

[**Saleh Albahli**](https://www.jmir.org/search?term=Saleh%20Albahli&type=author&precise=true) **(2021) et.al[16]**, proposed a diagnostic technique that classifies COVID-19 x-ray images from normal x-ray images and those specific to 14 other chest diseases. The deep learning models used in this study were pretrained on the ImageNet dataset, and transfer learning was used for fast training. Finally, concluded that the proposed pipeline can detect COVID-19 with a higher accuracy along with detecting 14 other chest diseases based on x-ray images. This is achieved by dividing the classification task into multiple steps rather than classifying them collectively.

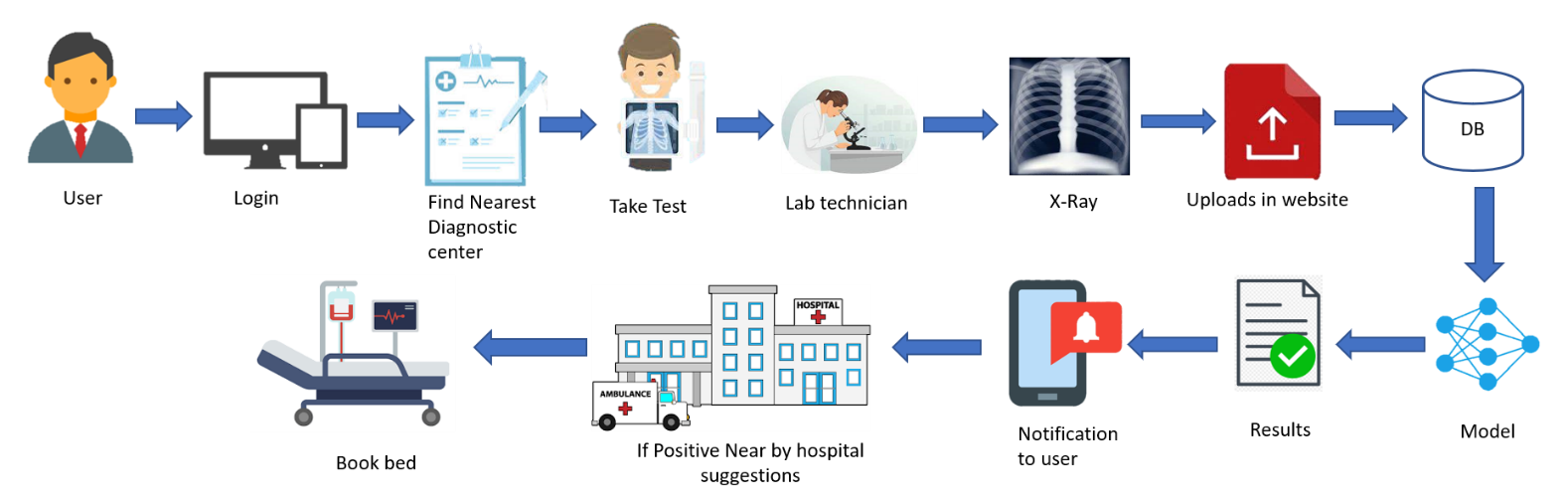
[**Nur-A- Alam**](https://sciprofiles.com/profile/author/a0JwRUovU1pGNkh6NitDM291cHFFZ2krMTBHSU53NnRLaTRwQlc4d09VWT0=) **(2021) et.al[17]**, proposed a machine vision approach to detect COVID-19 from the chest X-ray images. The proposed feature fusion system showed a higher classification accuracy (99.49%) than the accuracies obtained by using features obtained by individual feature extraction techniques, such as HOG and CNN. CNN produced the best classification accuracy compared to the other classification techniques, such as ANN, KNN and SVM.

**CHAPTER-3**

1. **METHODOLOGY**

**3.1 Proposed Work**

In this project, we have developed a complete end to end system which enables users to access the nearby diagnostics center and get instant results when the lab technician uploads the x-ray by logging into the technician account. Then covid-19 test result is updated in his profile and incase if user infected with covid, he will get nearby hospital suggestions which are dealing with covid and able to reserve the bed or ventilator etc.,

****

**Fig-3.1.1 Proposed system Architecture for Covid detection**

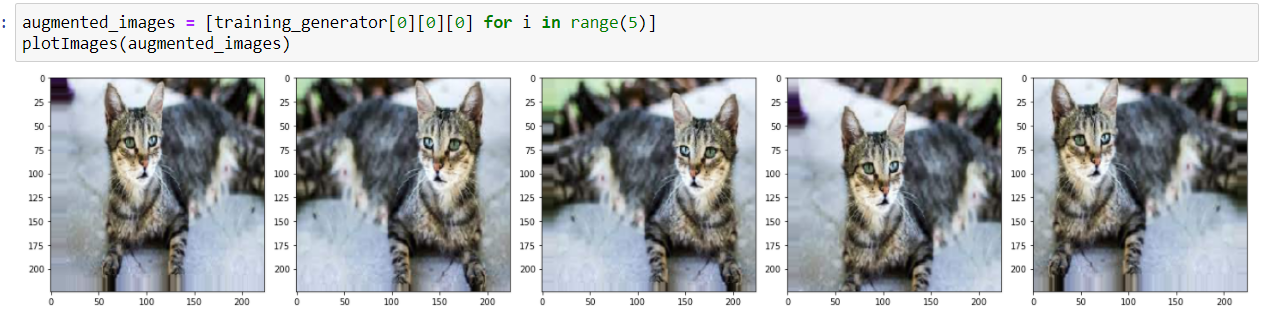
**3.2 Data Augmentation**

In the real world scenario, we may have a dataset of images taken in a limited set of conditions.But, our target application may exist in a variety of conditions to avoid the unknown scenarios for the model. We will apply data augmentation to the available data and create more variations in data.

Data Augmentation is a technique used to artificially increase dataset size. Take a sample from the dataset, modify it somehow, add it to the original dataset — and now your dataset is one sample larger.

There are a number of methods to apply augmentation below are some of the popular image augmentation.

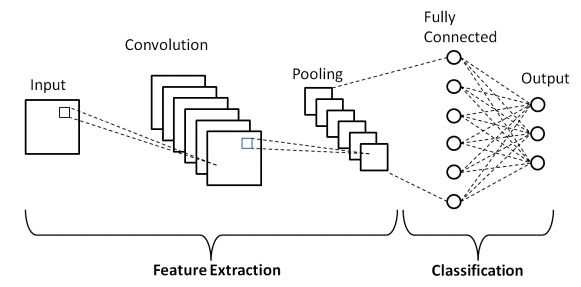
* Crop — takes a part of the images.
* Rotation — rotates the image around the center (or some other point).
* Flip — mirrors image around a horizontal or vertical line.
* The most popular Filters are blurring and sharpening. While blurring smoothes edges and details, sharpening highlights them.
* Affine Transformation- any transformations that preserve parallel lines.
* Adding Noise — such as blackening and whitening random pixels (salt & pepper noise), adding Gaussian noise, or even removing the whole region from an image (cutout).
* Color change makes the image darker or brighter, grayscale or extremely saturated, less or more contrasted.

****

**Fig-3.2.1 Example of Augmentation**

**3.3 Convolutional Neural Networks (CNN)**

Convolutional neural networks refer to a sub-category of neural networks. A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and are used mainly for image processing, classification, segmentation and also for other auto correlated data. A convolution is essentially sliding a filter over the input. There are four types of layers for a convolutional neural network



**Fig-3.3.1 Basic CNN Architecture**

1. Convolutional layer
2. Pooling layer
3. ReLU correction layer
4. Fully-connected layer.

**3.3.1 Convolution Layer**

Its purpose is to detect the presence of a set of features in the images received as input. This is done by convolution filtering: the principle is to “drag” a window representing the feature on the image, and to calculate the convolution product between the feature and each portion of the scanned image.

**3.3.2 Pooling Layer**

The pooling operation consists in reducing the size of the images while preserving their important characteristics.

**3.3.3 ReLU correction layer**

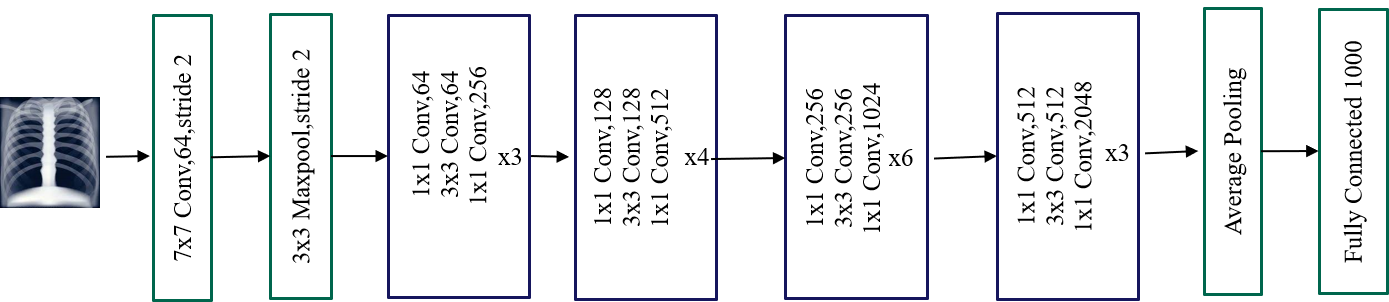
ReLU (Rectified Linear Units) refers to the real non-linear function defined by ReLU(x)=max(0,x). The ReLU correction layer replaces all negative values received as inputs by zeros. It acts as an activation function.

**3.3.4 Fully Connected Layer**

The last fully-connected layer classifies the image as an input to the network: it returns a vector of size N, where N is the number of classes in our image classification problem.

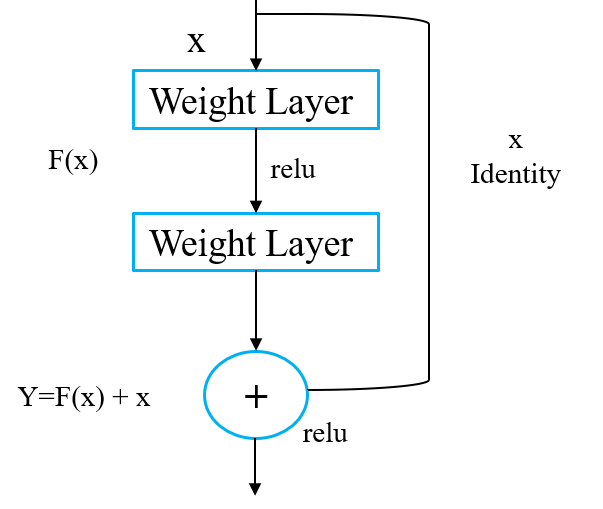
**3.4 Residual Neural Networks (ResNet)**

The ResNet model is a deep learning model developed by Microsoft Research Team that won the 2015 “ImageNet Large Scale Visual Recognition Challenge (ILSVRC)” competition with an error rate of 3.57%. Each layer of a ResNet consists of several blocks. With this model, when the residual layer structure is determined, the number of parameters calculated is reduced compared to other models.



**Fig-3.4.1 ResNet50 Architecture**

The tendency to add so many layers by deep learning practitioners is to extract important features from complex images. So, the first layers may detect edges, and the subsequent layers at the end may detect recognizable shapes, like tires of a car. But if we add more than 30 layers to the network, then its performance suffers and it attains a low accuracy. This is contrary to the thinking that the addition of layers will make a neural network better. This is not due to overfitting, because in that case, one may use dropout and regularization techniques to solve the issue altogether. It’s mainly present because of the popular vanishing gradient problem.



**Fig-3.4.2 Residual Block**

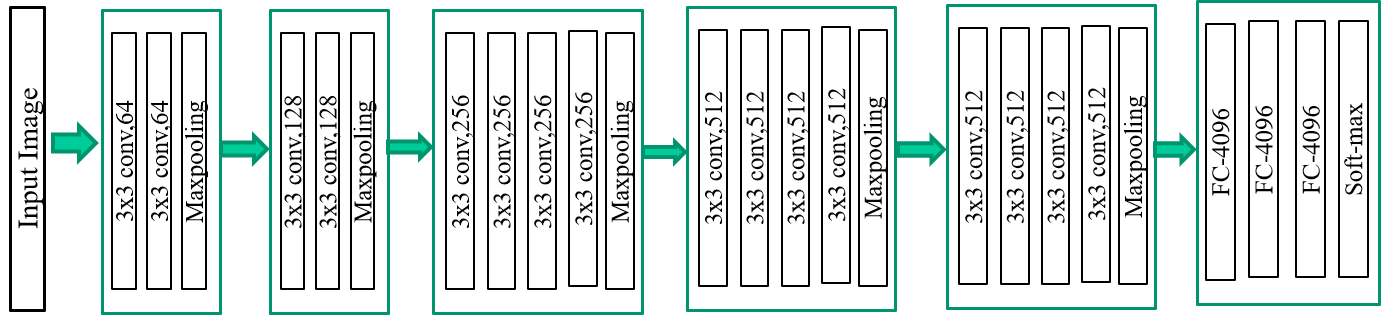
The output of the previous layer is added to the output of the layer after it in the residual block. The hop or skip could be 1, 2 or even 3.

When adding, the dimensions of x may be different than F(x) due to the convolution process, resulting in a reduction of its dimensions.

Thus, we add an additional 1 x 1 convolution layer to change the dimensions of x.

**3.5 Visual Geometry Group (VGG-19)**

It was proposed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University in 2014 in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”. The input to the network is image of dimensions (224, 224, 3).VGG-19 consists of 19 layers (16 convolution layers, 3 Fully connected layer, 5 MaxPool layers and 1 SoftMax layer). VGG has small filters 3x3 with more depth instead of having large filters. There are other variants of VGG like VGG11, VGG16 and others. VGG19 has 19.6 billion FLOPs.



**Fig-3.5.1 VGG-19 Architecture**

In both variations of VGGNet there consists of two Fully Connected layers with 4096 channels each which is followed by another fully connected layer with 1000 channels to predict 1000 labels. Last fully connected layer uses the softmax layer for classification purposes.

The first two layers are convolutional layers with 3\*3 filters, and first two layers use 64 filters that results in 224\*224\*64 volumes with the same convolutions are used. The filters are always 3\*3 with stride of 1

* After this, pooling layer was used with max-pool of 2\*2 size and stride 2 which reduces height and width of a volume from 224\*224\*64 to 112\*112\*64.
* This is followed by 2 more convolution layers with 128 filters. This results in the new dimension of 112\*112\*128.
* After pooling layer is used, volume is reduced to 56\*56\*128.
* Two more convolution layers are added with 256 filters each followed by down sampling layer that reduces the size to 28\*28\*256.
* Two more stacks each with 3 convolution layers are separated by a max-pool layer.
* After the final pooling layer, 7\*7\*512 volume is flattened into Fully Connected (FC) layer with 4096 channels and softmax output of 1000 classes.

**3.6 Inception-V3 Network**

Inception v3 is a [convolutional neural network](https://en.wikipedia.org/wiki/Convolutional_neural_network) for assisting in [image analysis](https://en.wikipedia.org/wiki/Image_analysis) and [object detection](https://en.wikipedia.org/wiki/Object_detection), and got its start as a module for [Googlenet](https://en.wikipedia.org/wiki/Googlenet). It is the third edition of Google's Inception Convolutional Neural Network, originally introduced during the ImageNet Recognition Challenge. Just as [ImageNet](https://en.wikipedia.org/wiki/ImageNet) can be thought of as a database of classified visual objects, Inception helps classification of objects in the world of [computer vision](https://en.wikipedia.org/wiki/Computer_vision).

The Inception-v3 network has 48 layers. Inception-v3 architecture makes several improvements including using Label Smoothing, Factorized 7 x 7 convolutions, and the use of an auxiliary classifier to propagate label information lower down the network (along with the use of batch normalization for layers in the side head).

**3.6.1 Factorizing Convolutions**

The aim of factorizing Convolutions is to reduce the number of connections/parameters without decreasing the network efficiency.

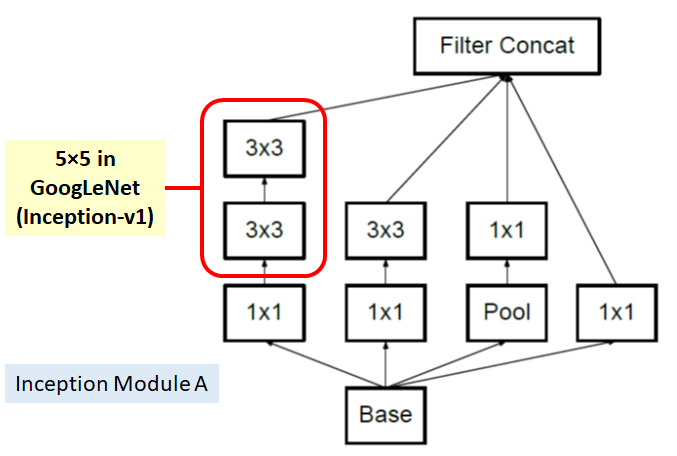
**3.6.2 Factorization Into Smaller Convolutions**

Two 3×3 convolutions replaces one 5×5 convolution as follows:

By using 1 layer of 5×5 filter, number of parameters = 5×5=25

By using 2 layers of 3×3 filters, number of parameters = 3×3+3×3=18

Number of parameters is reduced by 28%



**Fig-3.6.1 Inception Block-A**

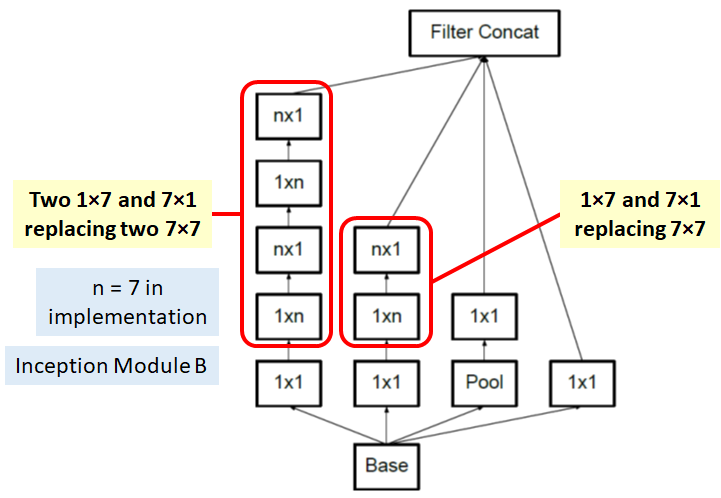
## 3.6.3 Factorization Into Asymmetric Convolutions

One 3×1 convolution followed by one 1×3 convolution replaces one 3×3 convolution as follows:

By using 3×3 filter, number of parameters = 3×3=9

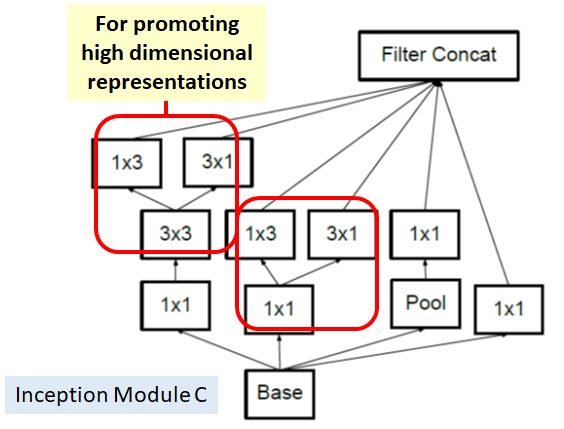
By using 3×1 and 1×3 filters, number of parameters = 3×1+1×3=6

Number of parameters is reduced by 33%

****

**Fig-3.6.2 Inception Block-B**

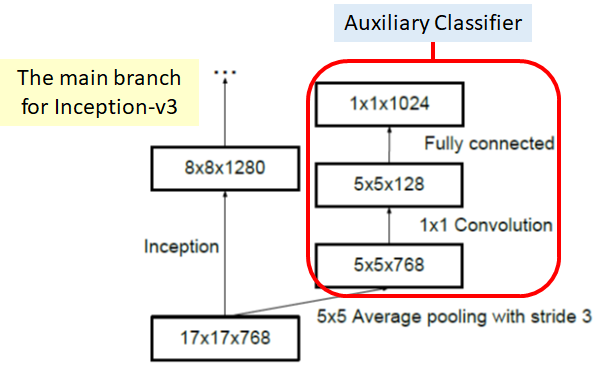
And Inception module C is also proposed for promoting high dimensional representations according to author descriptions as follows:

****

**Fig-3.6.3 Inception Block-C**

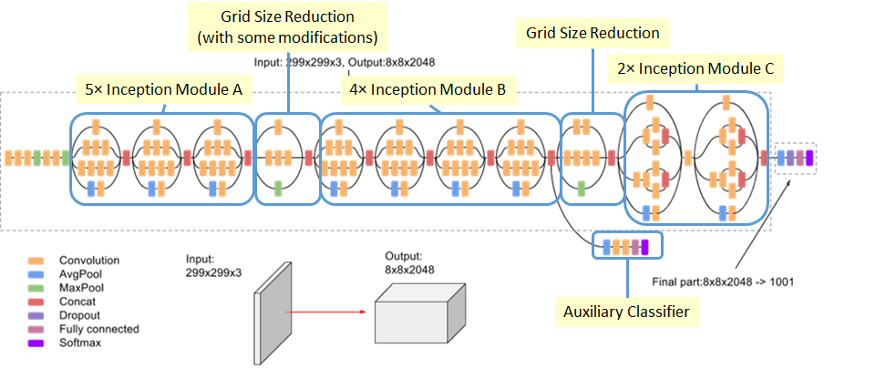
**3.6.4 Auxiliary Classifier**

Only 1 auxiliary classifier is used on the top of the last 17×17 layer, instead of using 2 auxiliary classifiers.

****

**Fig-3.6.4 Inception With Auxiliary Classifier**

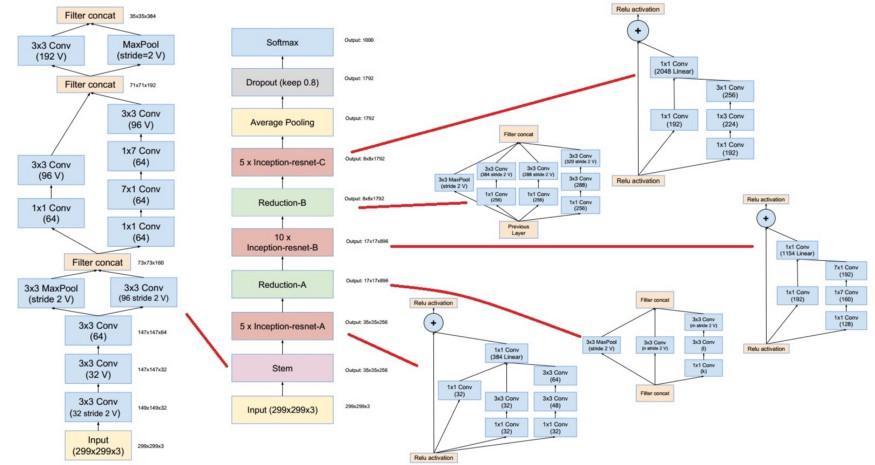
The purpose is also different. In GoogLeNet / Inception-v1, auxiliary classifiers are used for having deeper networks. In Inception-v3, the auxiliary classifier is used as a regularizer. So, actually, in deep learning, the modules are still quite intuitive.Batch normalization, suggested in Inception-v2, is also used in the auxiliary classifier.



**Fig-3.6.5 Inception-v3 Architecture**

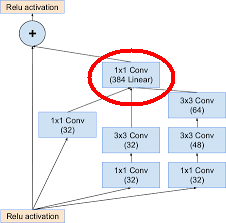
**3.7 Inception ResNet:**

Inception-ResNet-v2 is a variation of [Inception V3](http://arxiv.org/abs/1512.00567) model which borrows some ideas from Microsoft's ResNet papers. Inception-ResNet-v2 is a convolutional neural architecture that builds on the Inception family of architectures but incorporates [residual connections](https://paperswithcode.com/method/residual-connection) (replacing the filter concatenation stage of the Inception architecture). Residual connections allow shortcuts in the model and have allowed researchers to successfully train even deeper neural networks, which have led to even better performance. This has also enabled significant simplification of the Inception blocks. The Inception-ResNet-v2 architecture is more accurate than previous state of the art models, which reports the Top-1 and Top-5 validation accuracies on the [ILSVRC 2012 image classification benchmark](http://image-net.org/challenges/LSVRC/2012/) based on a single crop of the image. And this new model only requires roughly twice the memory and computation compared to Inception V3.



**Fig-3.7.1 Inception ResNet Architecture**

**3.7.1 Residual Inception blocks**

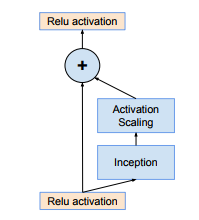


**Fig-3.7.2 Residual Inception Blocks**

1. Each Inception block is followed by a filter expansion layer (1 × 1 convolution without activation) which is used for scaling up the dimensionality of the filter bank before the addition to match the depth of the input.
2. In the case of Inception-ResNet, batch-normalization is used only on top of the traditional layers, but not on top of the summations.

**3.7.2 Scaling of Residuals**

According to the authors, if the number of filters exceeded 1000, the residual variants started to exhibit instabilities and the network had just “died” early in the training, meaning that the last layer before the average pooling started to produce only zeros after a few tens of thousands of iterations. This could not be prevented, neither by lowering the learning rate, nor by adding an extra batch normalization to this layer.

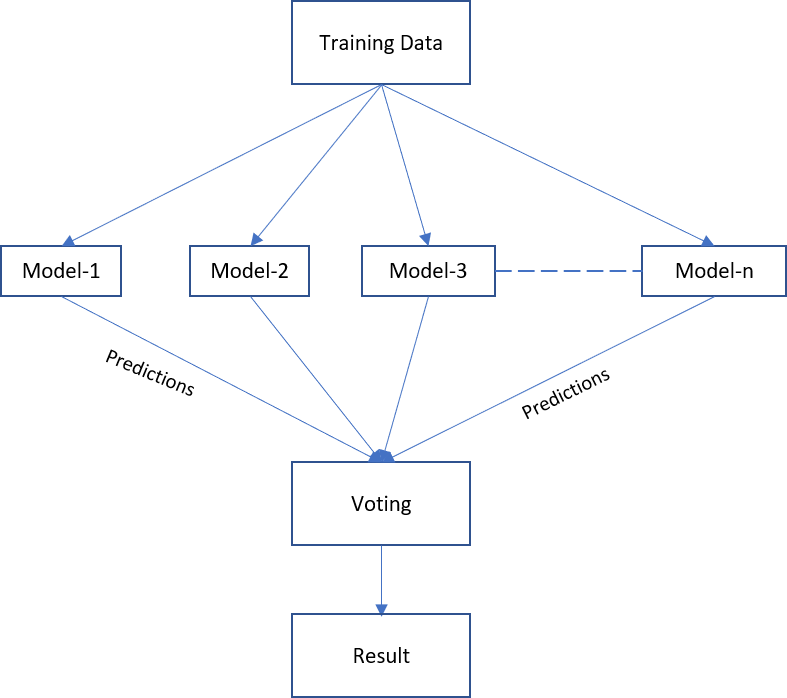


**Fig-3.7.3 Scaling of Residuals**

According to them, scaling down the residuals before adding them to the previous layer activation seemed to stabilize the training. To scale the residuals, scaling factors between **0.1** and **0.3** were picked.

**3.8 Ensemble Leaning**

Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one. Other applications of ensemble learning include assigning a confidence to the decision made by the model, selecting optimal (or near optimal) features, data fusion, incremental learning, nonstationary learning and error-correcting. This article focuses on classification related applications of ensemble learning, however, all principle ideas described below can be easily generalized to function approximation or prediction type problems as well.An ensemble-based system is obtained by combining diverse models (henceforth classifiers). Therefore, such systems are also known as multiple classifier systems, or just ensemble systems. There are several scenarios where using an ensemble based system makes statistical sense, which are discussed below in detail. However, in order to fully and practically appreciate the importance of using multiple classifier systems, it is perhaps instructive to look at a psychological backdrop to this otherwise statistically sound argument: we use such an approach routinely in our daily lives by asking the opinions of several experts before making a decision.



**Fig-3.8.1 Ensemble model**

**3.9 HTML**

HTML stands for Hyper Text Markup Language. It is the standard markup language for creating Web pages.And it describes the structure of a Web page.This consists of a series of elements.HTML elements tell the browser how to display the content.HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

**3.10 CSS**

**CSS** is the acronym for **"Cascading Style Sheet '',** is a simple design language intended to simplify the process of making web pages presentable. It is used to control the style of a web document in a simple and easy way. CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

**3.11 Bootstrap**

[Bootstrap](http://getbootstrap.com/) is a powerful toolkit - a collection of HTML, CSS, and JavaScript tools for creating and building web pages and web applications.Web designers and web [developers like Bootstrap](https://www.toptal.com/bootstrap) because it is flexible and easy to work with. Its main advantages are that it is responsive by design, it maintains wide browser compatibility, it offers consistent design by using reusable components, and it is very easy to use and quick to learn. It offers rich extensibility using JavaScript, coming with built-in support for jQuery plugins and a programmatic JavaScript API. Bootstrap can be used with any IDE or editor, and any server side technology and language, from ASP.NET to PHP to Ruby on Rails.

The Bootstrap structure is pretty simple and self-explanatory. It includes precompiled files that enable quick usage in any web project. Besides compiled and minified CSS and JS files, it also includes fonts from [Glyphicons](http://glyphicons.com/), and the optional starting Bootstrap theme.

**3.12 DJango**

Django is an extremely popular and fully featured server-side web framework, written in Python. It is a Python-based web framework that allows you to quickly create efficient web applications. It is also called batteries included framework because Django provides built-in features for everything including Django Admin Interface, default database – SQLlite3, etc. When you’re building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc. Django gives you ready-made components to use and that too for rapid development.

Django development environment consists of installing and setting up Python, Django, and a Database System. Since Django deals with web applications, you would need a web server setup as well.

## Step 1 – Installing Python

Django is written in 100% pure Python code, so you'll need to install Python on your system. Latest Django version requires Python 2.6.5 or higher.

**Step 2 - Installing Django**

Installing Django is very easy, but the steps required for its installation depends on your operating system. Since Python is a platform-independent language, Django has one package that works everywhere regardless of your operating system.

### Windows Installation

First, PATH verification.

On some version of windows (windows 7) you might need to make sure the Path system variable has the path the following

C:\Python34\;C:\Python34\Lib\site-packages\django\bin\

Then, extract and install Django.

c:/>cd c:\Django-x.xx

Next, install Django by running the following command for which you will need administrative privileges in windows shell "cmd"

c:\Django-x.xx>python setup.py install

To test your installation, open a command prompt and type the following command

c:\>python -c “import django;print(django.get\_version())”

## Step 3 – Database Setup

Django supports several major database engines and you can set up any of them.

* [MySQL](http://www.mysql.com/)
* [PostgreSQL](http://www.postgresql.org/)
* [SQLite 3](http://www.sqlite.org/)
* [Oracle](http://www.oracle.com/index.html)
* [MongoDb](https://django-mongodb-engine.readthedocs.org/en/latest/)
* [GoogleAppEngine Datastore](https://cloud.google.com/appengine/articles/django-nonrel)

## Step 4 – Web Server

Django comes with a lightweight web server for developing and testing applications. This server is pre-configured to work with Django, and more importantly, it restarts whenever you modify the code. However, Django does support Apache and other popular web servers such as Lighttpd.

**3.13 IBM Cloud**

IBM Cloud is a set of [cloud computing](https://en.wikipedia.org/wiki/Cloud_computing) services for business offered by the information technology company [IBM](https://en.wikipedia.org/wiki/IBM). It combines [platform as a service](https://en.wikipedia.org/wiki/Platform_as_a_service) (PaaS) with [infrastructure as a service](https://en.wikipedia.org/wiki/Infrastructure_as_a_service) (IaaS). The platform scales and supports both small development teams and organizations, and large enterprise businesses. It is globally deployed across [data centers](https://en.wikipedia.org/wiki/Data_centers) around the world. IBM's main competitors in the cloud computing market include [Amazon Web Services](https://en.wikipedia.org/wiki/Amazon_Web_Services), [Microsoft Azure](https://en.wikipedia.org/wiki/Microsoft_Azure) and [Google Cloud Platform](https://en.wikipedia.org/wiki/Google_Cloud_Platform).

IBM Cloud offers the most open and secure public cloud platform for business, a next-generation hybrid multi cloud platform, advanced data and AI capabilities, and deep enterprise expertise across 20 industries. IBM Cloud hybrid cloud solutions deliver flexibility and portability for both applications and data. Linux®, Kubernetes, and containers support this hybrid cloud stack, and combine with RedHat® OpenShift® to create a common platform connecting on-premises and cloud resources.

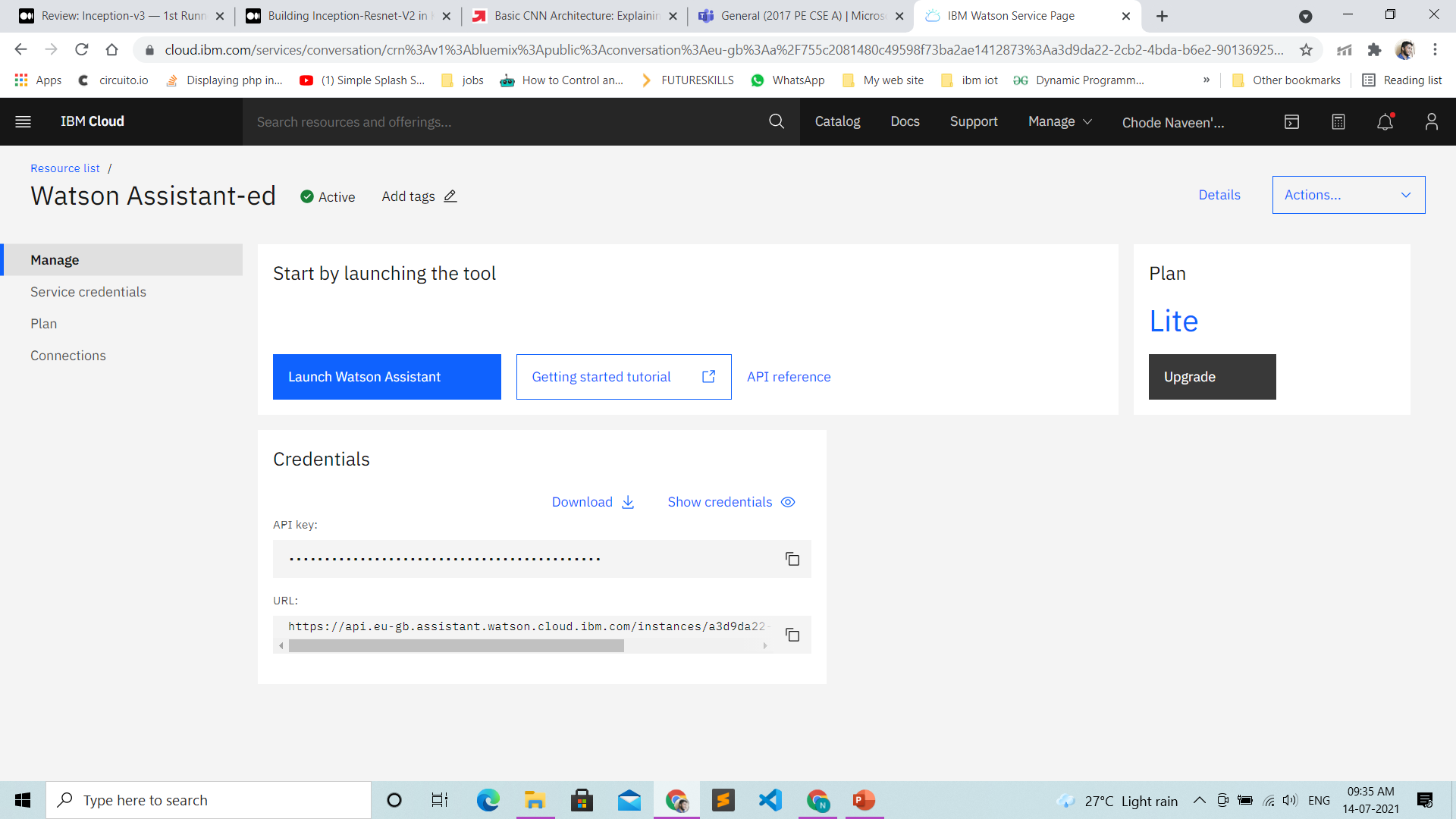
**3.13.1 IBM Watson**

Watson was created as a [question answering](https://en.wikipedia.org/wiki/Question_answering) (QA) computing system that IBM built to apply advanced [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing), [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval), [knowledge representation](https://en.wikipedia.org/wiki/Knowledge_representation), [automated reasoning](https://en.wikipedia.org/wiki/Automated_reasoning), and [machine learning](https://en.wikipedia.org/wiki/Machine_learning) technologies to the field of [open domain question answering](https://en.wikipedia.org/wiki/Open_domain_question_answering).[[2]](https://en.wikipedia.org/wiki/Watson_(computer)#cite_note-ibm-2)

When created, IBM stated that, "more than 100 different techniques are used to analyze natural language, identify sources, find and generate hypotheses, find and score evidence, and merge and rank hypotheses."

In recent years, the Watson capabilities have been extended and the way in which Watson works has been changed to take advantage of new deployment models (Watson on IBM Cloud) and evolved machine learning capabilities and optimised hardware available to developers and researchers. It is no longer purely a question answering (QA) computing system designed from Q&A pairs but can now 'see', 'hear', 'read', 'talk', 'taste', 'interpret', 'learn' and 'recommend'.

IBM’s portfolio of business-ready tools, applications and solutions, designed to reduce the costs and hurdles of AI adoption while optimizing outcomes and responsible use of AI. Watson advances trust from principle to practice. Transparent processes provide insight into AI-led decisions. We enable data privacy, compliance and security across highly regulated industries, and support an open, diverse ecosystem driving responsible use of AI.



**Fig-3.13.1.1 IBM Watson**

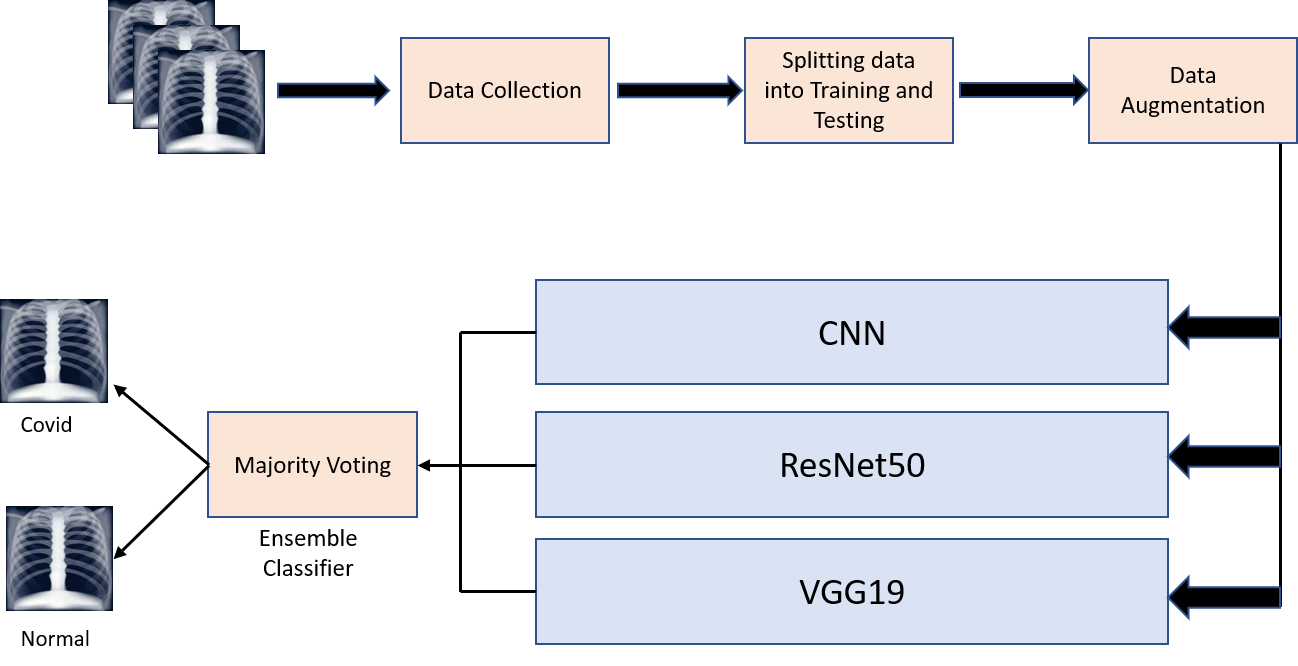
**CHAPTER - 4**

**4. IMPLEMENTATION**

In this project we have trained CNN, Resnet50, Vgg19, InceptionV3 and Inception Resnet. We trained deep learning models with our dataset mentioned earlier and tested them. Based on the results after training and testing we consider that Resnet50, Vgg19 and CNN are performing better among the all models we have trained.

We have chosen voting techniques to improve the performance by ensemble the models resnet50,vgg19 and CNN.

The model is used to predict whether the person whose X-Ray has been given is effected by covid19 or not.

****

**Fig-4.0.1 Proposed Ensemble architecture**

**4.1. Dataset**

As known, it is very difficult to find an open source data set since COVID-19 is a new disease type.

We found a data set with a total of 6900 samples.

Where the data is categorized into three classes covid, normal and pneumonia. 2300 samples were there in each class.

As per our requirement we have taken only covid and normal samples and divided them into training and testing datasets.

•Training : 2000(covid)+2000(normal)=4000

•Testing : 300(covid)+300(normal)=600

**Source Link :** <https://www.kaggle.com/amanullahasraf/covid19-pneumonia-normal-chest-xray-pa-dataset?select=pneumonia>

**4.1.1 Covid X-ray Images**

****

**Fig-4.1.1 Covid X-ray Images**

**4.1.2 Normal X-ray Images**

****

**Fig-4.1.2 Normal X-ray Images**

**4.2 Tools and Modules**

* Python
* Jupyter notebook
* Anaconda
* Django
* Ibm Watson
* Visual studio

**4.3 Source Code**

**4.3.1 Data splitting**

import os

import numpy as np

import shutil

rootdir= 'D:/Covid19 papers/Covid19 Detection/CovidCheastXray' #path of the original folder

classes = ['COVID','Normal']

for i in classes:

os.makedirs(rootdir +'/train/' + i)

os.makedirs(rootdir +'/test/' + i)

source = rootdir + '/' + i

allFileNames = os.listdir(source)

np.random.shuffle(allFileNames)

test\_ratio = 0.25

train\_FileNames, test\_FileNames = np.split(np.array(allFileNames),

[int(len(allFileNames)\* (1 - test\_ratio))])

train\_FileNames = [source+'/'+ name for name in train\_FileNames.tolist()]

test\_FileNames = [source+'/' + name for name in test\_FileNames.tolist()]

for name in train\_FileNames:

shutil.copy(name, rootdir +'/train/' + i)

for name in test\_FileNames:

shutil.copy(name, rootdir +'/test/' + i)

**4.3.2 Data Augmentation**

training\_data\_generator = ImageDataGenerator(rescale=1./255,width\_shift\_range=0.1, height\_shift\_range=0.1, shear\_range=0.1, zoom\_range=0.1, horizontal\_flip=True)

def plotImages(images\_arr):

fig, axes = plt.subplots(1, 5, figsize=(20,20))

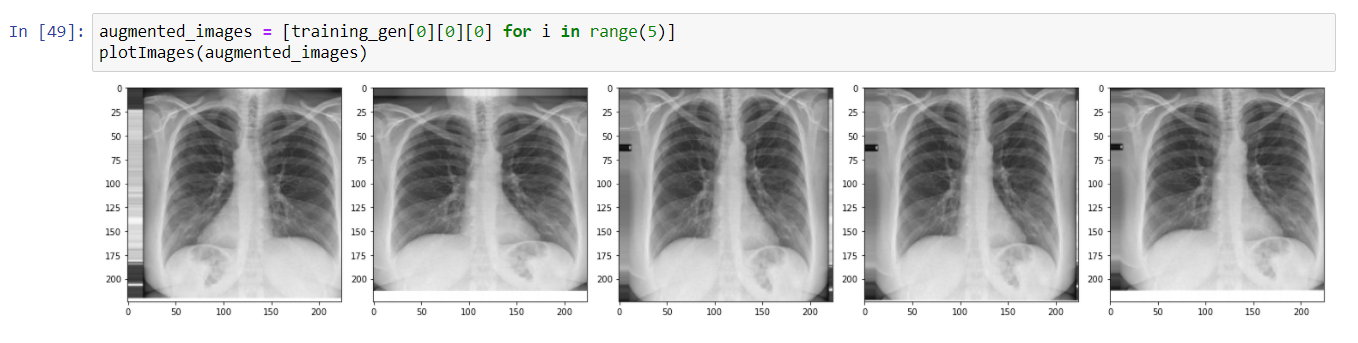
axes = axes.flatten()

for img, ax in zip(images\_arr, axes):

ax.imshow(img)

plt.tight\_layout()

plt.show()

****

**Fig-4.3.2.1 Augmented Images**

**4.3.3 CNN**

def ownmodel():

model = Sequential()

model.add(Conv2D(32, kernel\_size=(3,3),activation='relu',padding='same',input\_shape=(224,224,3)))

model.add(BatchNormalization(axis=-1))

model.add(MaxPooling2D((2, 2), padding='same'))

model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))

model.add(BatchNormalization(axis=-1))

model.add(MaxPooling2D((2, 2), padding='same'))

model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))

model.add(BatchNormalization(axis=-1))

model.add(MaxPooling2D((2, 2), padding='same'))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(256, activation='relu'))

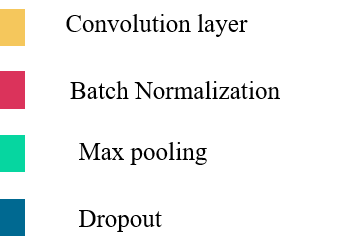
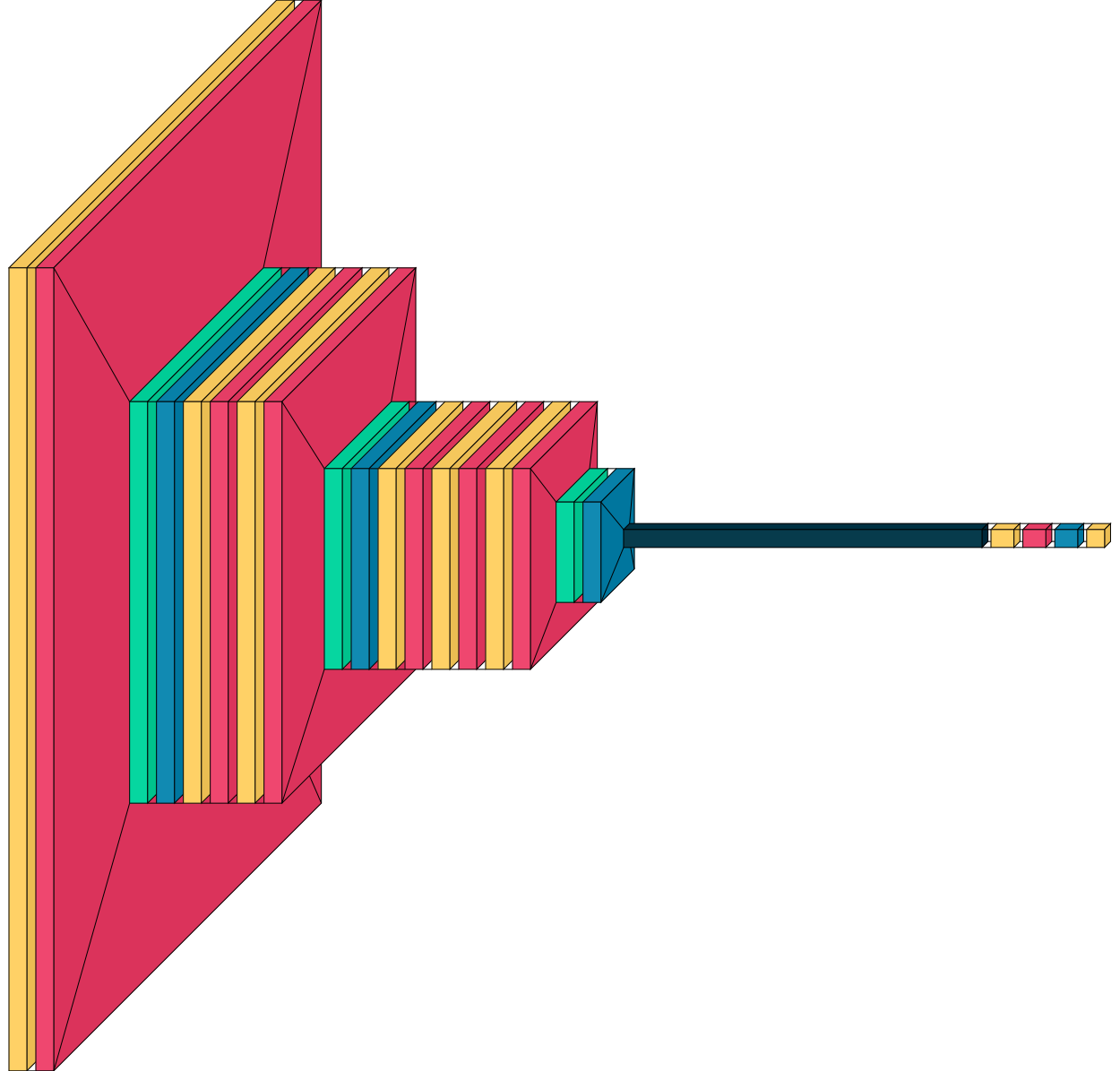
model.add(BatchNormalization())

model.add(Dropout(0.5))

model.add(Dense(2, activation='softmax'))

model.compile(loss=keras.losses.binary\_crossentropy,optimizer='adam',metrics=['accuracy'])

return model

****

**Fig-4.3.3.1 Custom CNN**

**4.3.4 ResNet50**

from keras.applications.resnet50 import ResNet50

%matplotlib inline

base\_model=ResNet50(weights='imagenet',include\_top=False)

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(1024, activation='relu')(x)

predictions = Dense(2, activation='softmax')(x)

model = Model(inputs=base\_model.input, outputs=predictions)

model.compile(optimizer=Adam(lr=learning, beta\_1=0.9, beta\_2=0.999, amsgrad=False), loss='categorical\_crossentropy',metrics=['accuracy'])

import warnings

warnings.filterwarnings("ignore")

model.summary()

**4.3.5 VGG19**

from tensorflow.keras.applications.vgg19 import VGG19

from tensorflow.keras.preprocessing import image

from tensorflow.keras.applications.vgg19 import preprocess\_input

from tensorflow.keras.models import Model

import numpy as np

%matplotlib inline

base\_model=tf.keras.applications.vgg19.VGG19(weights='imagenet',include\_top=False)

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(1024, activation='relu')(x)

predictions = Dense(2, activation='softmax')(x)

model = Model(inputs=base\_model.input, outputs=predictions)

model.compile(optimizer=Adam(lr=learning, beta\_1=0.9, beta\_2=0.999, amsgrad=False), loss='categorical\_crossentropy',metrics=['accuracy'])

import warnings

warnings.filterwarnings("ignore")

model.summary()

**4.3.6 Model Checkpoints**

checkpoint\_filepath = "weights-improvement-{epoch:02d}-{val\_accuracy:.2f}.h5"

model\_checkpoint\_callback = tf.keras.callbacks.ModelCheckpoint(

filepath=checkpoint\_filepath,

save\_weights\_only=False,

monitor='val\_accuracy',

mode='max',

save\_best\_only=True)

**4.3.7 Model Training**

H = model.fit\_generator( training\_generator, steps\_per\_epoch=len(training\_generator.filenames) // BATCH\_SIZE, epochs=50,

validation\_data=validation\_generator,

validation\_steps=len(validation\_generator.filenames)//BATCH\_SIZE, callbacks=[model\_checkpoint\_callback,PlotLossesKeras(), CSVLogger(TRAINING\_LOGS\_FILE, append=False, separator=";")], verbose=1)

**4.3.8 Performance Analysis**

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_curve, auc

LABELS = ["covid-19","normal"]

def show\_confusion\_matrix(validations, predictions):

matrix = confusion\_matrix(validations,predictions)

plt.figure(figsize=(8, 6))

sns.heatmap(matrix,

cmap="coolwarm",

linecolor='white',

linewidths=1,

xticklabels=LABELS,

yticklabels=LABELS,

annot=True,

fmt="d")

plt.title("Confusion Matrix")

plt.ylabel("True Label")

plt.xlabel("Predicted Label")

plt.show()

validation\_generator = validation\_data\_generator.flow\_from\_directory(

validation\_data\_dir,

target\_size=(224, 224),

batch\_size=64,

class\_mode="categorical",

shuffle=False)

filenames = validation\_generator.filenames

nb\_samples = len(filenames)

Y\_pred = model.predict\_generator(validation\_generator,(nb\_samples//BATCH\_SIZE))

y\_pred = np.argmax(Y\_pred, axis=1)

print('Confusion Matrix')

show\_confusion\_matrix(validation\_generator.classes, y\_pred)

print(confusion\_matrix(validation\_generator.classes, y\_pred))

print('Classification Report')

target\_names = ["covid-19","normal"]

print(classification\_report(validation\_generator.classes, y\_pred, target\_names=target\_names))

# Plot linewidth.

lw = 2

# Compute ROC curve and ROC area for each class

fpr = dict()

tpr = dict()

roc\_auc = dict()

for i in range(2):

fpr[i], tpr[i], \_ = roc\_curve(validation\_generator.classes, y\_pred)

roc\_auc[i] = auc(fpr[i], tpr[i])

plt.figure()

lw = 2

plt.plot(fpr[0], tpr[0], color='darkorange',

lw=lw, label='ROC curve (area = %0.4f)' % roc\_auc[1])

plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')

plt.xlim([-0.01, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver operating characteristic example')

plt.legend(loc="lower right")

plt.show()

**4.3.9 Ensemble with Voting**

from tensorflow.keras.preprocessing import image

testimg1=image.load\_img("D:/4.2/Mainproject/cov/validation/covid/13.jpeg",target\_size=(224, 224))

testimg=image.img\_to\_array(testimg1)

testimg=np.expand\_dims(testimg,axis=0)

testimg= testimg/255.0

result1=model1.predict(testimg)

print(result1)

l=np.argmax(result1[0])

m1=l

result1=model2.predict(testimg)

print(result1)

l=np.argmax(result1[0])

m2=l

result1=model3.predict(testimg)

print(result1)

l=np.argmax(result1[0])

m3=l

v=[m1,m2,m3]

if v.count(0)>v.count(1):

r=0

else:

r=1

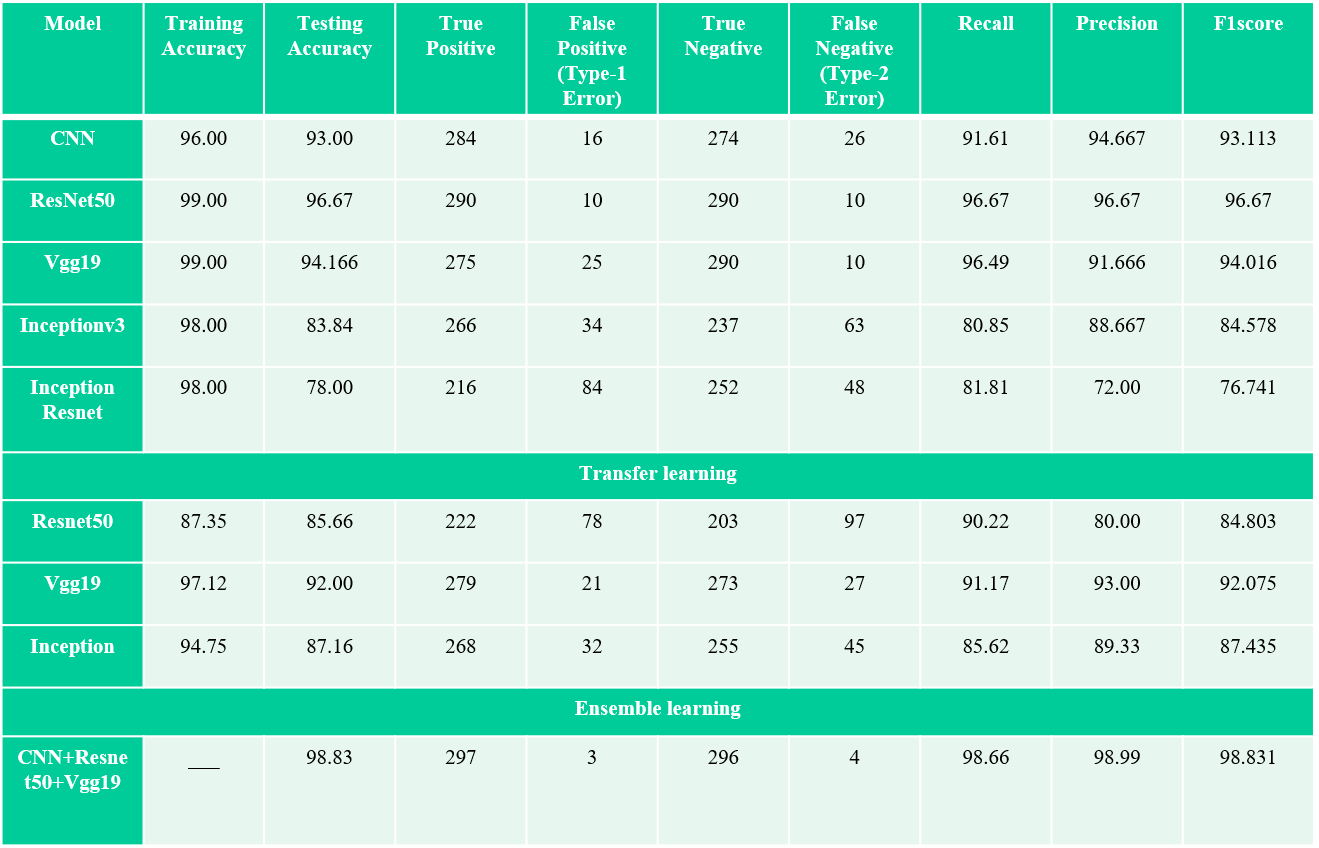
print(labels[r])

**CHAPTER-5**

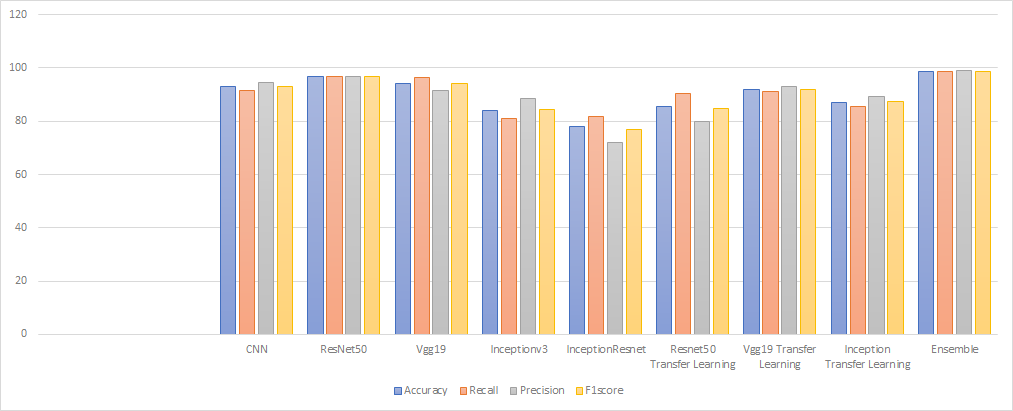
**5. RESULTS**

**5.1 Performance of Models**

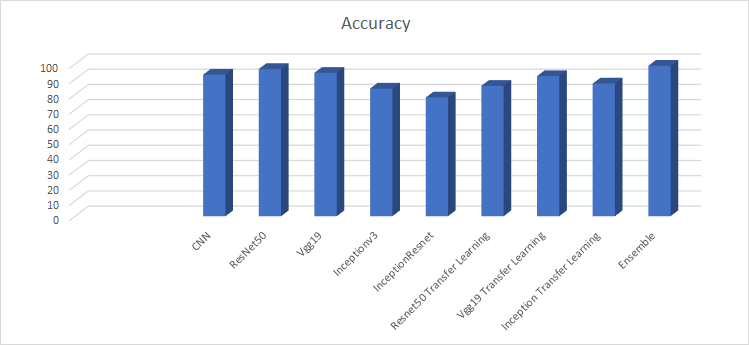
We have tested different deep learning models which are trained with a dataset of 2000 covid effected x-ray images and 2000 normal x-ray images. The testing dataset of 600 samples. We have considered some Traditional metrics like True Positives, False Positives, Accuracy, Precision, F1 score. The following table 5.1.1 illustrates the performances of different models.



**Table-5.1.1 Performance of Models**

****

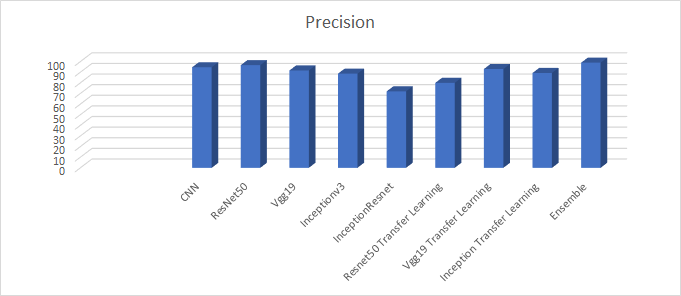
**Fig-5.1.2 Performance Analysis of individual models**

****

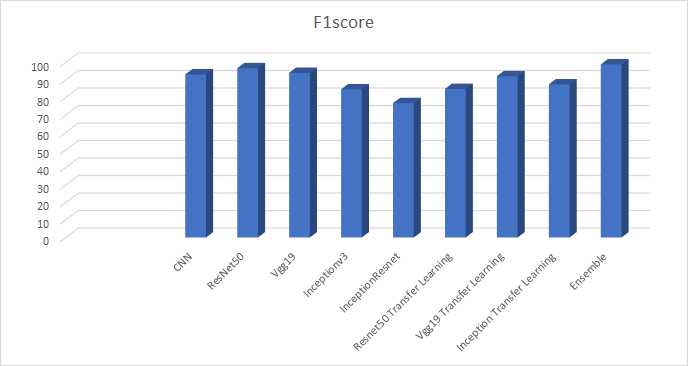
**Fig-5.1.3 Comparison of Accuracy obtained from trained models**

****

**Fig-5.1.4 Comparison of Recall obtained from trained models**

****

**Fig-5.1.5 Comparison of Precision obtained from trained models**

****

**Fig-5.1.6 Comparison of F1-score obtained from trained models**

**5.2 OUTPUT**

**CHAPTER - 6**

**6.CONCLUSIONS AND FUTURE SCOPE**

In this project, we have developed a complete end to end system which enables user to access the nearby diagnostics center and get instant results when the lab technician uploads the X Ray.Then covid-19 test result is updated in his profile and incase if user infected with covid, he will get nearby hospital suggestions. And for the detection of covid-19, we have trained the dataset with CNN and advanced CNN models. Data augmentation is also applied on the original training images to enhance the regularization of the model . In order to improve the performance of the model we have ensemble the models which have better performance i.e., Resnet50, VGG19, CNN. We used majority voting technique to classify the final result as Covid or Normal. When we look at the results obtained, 98.83 of overall accuracy is obtained as a result of combining the better performed individual models into one. Since the approach was proven to be reliable by considering different criteria, it is predicted that it can be used to provide another idea for experts during the diagnosis of COVID-19 disease. Along with this, we have created a chat bot to assist the user while using our web application which is useful to know basic information about the covid. In order to contribute to this field in future studies, the plan is to continue studies using image processing and different deep learning models.

**CHAPTER-7**

**7. REFERENCES**

1.Paluru, N., Dayal, A., Jenssen, H. B., Sakinis, T., Cenkeramaddi, L. R., Prakash, J., & Yalavarthy, P. K. (2021). Anam-Net: Anamorphic depth embedding-based lightweight CNN for segmentation of anomalies in COVID-19 chest CT images. IEEE Transactions on Neural Networks and Learning Systems, 32(3), 932-946. (Base Paper )

2.Canayaz, M. (2021). MH-COVIDNet: Diagnosis of COVID-19 using deep neural networks and meta-heuristic-based feature selection on X-ray images. Biomedical Signal Processing and Control, 64, 102257.

3.Ezzat, D., Hassanien, A. E., & Ella, H. A. (2020). An optimized deep learning architecture for the diagnosis of COVID-19 disease based on gravitational search optimization. Applied Soft Computing, 106742.

4.Tabik, S., Gómez-Ríos, A., Martín-Rodríguez, J. L., Sevillano-García, I., Rey-Area, M., Charte, D., ... & Herrera, F. (2020). COVIDGR dataset and COVID-SDNet methodology for predicting COVID-19 based on Chest X-Ray images. IEEE journal of biomedical and health informatics, 24(12), 3595-3605.

5. Wu, Y. H., Gao, S. H., Mei, J., Xu, J., Fan, D. P., Zhang, R. G., & Cheng, M. M. (2021). Jcs: An explainable covid-19 diagnosis system by joint classification and segmentation. IEEE Transactions on Image Processing, 30, 3113-3126.

6. BhawnaNigama,AyanNigam,RahulJain, ShubhamDodia, NidhiArora, B.Annappa(2021), COVID-19: Automatic detection from X-ray images by utilizing deep learning methods, Volume 176,114883.