Emerging Methods For Early Detection Of Forest Fires

A PROJECT REPORT

Submitted by

Team ID: PNT2022TMID32061

Team Leader	**Ashok R**	**731619205002**
Team Member 1	**Brethika R D**	**731619205006**
Team Member 2	**Kabish S**	**731619205020**
Team Member 3	**Krithick sanjay K**	**731619205027**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

K S R INSTITUTE FOR ENGINEERING AND

TECHNOLOGY,

TIRUCHENGODE

ANNA UNIVERSITY: CHENNAI 600 025

INDEX

Chapter-1

INTRODUCTION

1.1Project Overview

The ecological balance is maintained by the forests.It acts as an environment that enrich the diversity of various organisms.The motive of the project is to detect the forest fire as early as possible so that we can preserve the life of various species prevailing in it from the fire. Utilizing the currently available techniques of smokesensors put in the buildings, fire detection can be incredibly challenging. Due to their outdated technology and design, they are costly and slow. The use of artificial intelligence for identification and issuing alerts with video from CCTV footages is critically examined in this study. For this project, a self-built dataset of videoframes with fire is used. The data is then preprocessed and a machine learning model is built using CNN. The dataset's test set is used as input to verify the method, and experiments are recorded. The goal of the project is to create a machine that is both affordable and very precise and can be applied to practically any fire detecting situation.

1.2 Purpose

One of the key elements in keeping the ecological in balance is forests. When a fire breaks out in a forest, it can be very dangerous. However, a forest fire is typically discovered after it has spread across a significant area. It might not always be able to put out the fire. As a result, environmental impact is worse than anticipated. The environment suffers because of the forest fire's large-scale carbon dioxide (CO_2) emissions. It would result in the global extinction of rare species. Additionally, it may have an effect on the weather, which may lead to serious problems like earthquakes, excessive rain, floods, and so forth.The forest is a big surface area covered with trees, tonnes of dried leaves, woodlands, and other things. When the fire first ignites, these substances help it grow. Fire might start for a variety of causes, including smoking, fireworks-themed events, or high summer temperatures. Once a fire starts, it won't stop until it has entirely burned itself out. When the fire is noticed as early as feasible, the damage and the cost associated with identifying it due to a forest fire can be minimised. Therefore, in this case, fire detection is crucial. A good effect can be had by locating the fire's specific location and notifying the fire authorities as soon as the fire occurs.Thus it is crucial to implement a system to identify fires as soon as possible.

Chapter 2

LITERATURE SURVEY

2.1 Existing problem

Smoke alarms and heat alarms are being used to detect fires. One module is not enough to monitor all of the potential fire prone areas, which is the fundamental drawback of smoke sensor alarms and heat sensor alarms. The only way to avoid a fire is to exercise caution at all times. Even if they are installed in every nook and cranny, it simply is not enough to constantly produce an efficient output. As the number of smoke sensors required rises, the price will rise by a factor of multiples. Within seconds of an accident or a fire, the suggested system can generate reliable and highly accurate alerts. One piece of software powers the entire surveillance network, which lowers costs. Data scientists and machine learning experts are actively conducting research in this area.

2.2 References

[1] Official webpage of the European Forest Fire Information System at: http://effis.jrc.ec.europa.eu/

[2] Official webpage of the Copernicus Earth Observation Programme at: http://www.copernicus.eu

[3] Forest Fires in Europe, Middle East and North Africa 2016, JRC Science for policy report, BN 978-92-79-71292-0, ISSN 1831-9424, doi:10.2760/17690, availabe at:
http://effis.jrc.ec.europa.eu/media/cms_page_media/40/Forest_fires_in_Europe_Middle_east_and_North_Africa_2016_final_pdf_JZU7HeL.pdf

[4] The 2018 Attica wildfires Wikipedia webpage available at: https://en.wikipedia.org/wiki/2018_Attica_wildfires

[5] László Földi and Rajmund Kuti, Characteristics of Forest Fires and their Impact on the Environment, Academic and Applied Research in
Military and Public Management Science (AARMS), Vol. 15, No. 1,
2016, pp. 5–17, ISSN 2064-0021;

[6] https://www.firedex.com

[7] Wolfgang Jendsch, Aerial Firefighting, Schiffer Publishing, 352 pp,
ISBN 9780764330681

[8] Chi Yuan, Youmin Zhang and Zhixiang Liu, A Survey on
Technologies for Automatic Forest Fire Monitoring, Detection and

Fighting Using UAVs and Remote Sensing Techniques, Canadian Journal of Forest Research, doi: 10.1139/cjfr-2014-0347
[9] W. Krüll, R. Tobera, I. Willms, H. Essen and N. V. Wahl, Early Forest Fire Detection and Verification Using Optical Smoke, Gas and Microwave Sensors, Procedia Engineering vol. 45, pp. 584-594, 2012 [10] Project SFEDA page at: http://www.interreg-balkanmed.eu/approved-project/22/
[11] V. C. Moulianitis, G. Thanellas, N. Xanthopoulos, N. A. Aspragathos, Evaluation of UAV based schemes for forest fire monitoring, 27th International Conference on Robotics in Alpe-Adria-Danube Region, RAAD 2018
[12] Official webpage of the ALTi Transition VTOL UAV at: https://www.altiuas.com/transition/
[13] Official webpage of the DJI Matrice 600 Pro UAV at: https://www.dji.com/matrice600-pro
[14] Official webpage of the DJI Matrice 200 series of UAVs at: https://www.dji.com/matrice-200-series
[15] Official webpage of Movidius at: https://www.movidius.com [16] Official webpage of the IMST iC880A LoRaWAN concentrator at: https://wireless-solutions.de/products/long-range-radio/ic880a.html

2.3 Problem Statement Definition

Forest fires result in a wide range of negative effects, including the destruction of wildlife habitat, the extinction of plants and animals, the destruction of nutrient-rich top soil, the reduction of forest cover, the loss of valuable timber resources, the ozone layer being destroyed, the loss of livelihood for tribal and poor people, the acceleration of global warming, the increase in atmospheric carbon dioxide concentration, the degradation of catchment areas, the loss of biodiversity, the spread of disease, etc. Thus, Develop a system to detect forest fires at the earliest stage as possible using the lastest technologies.

Chapter 3
IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

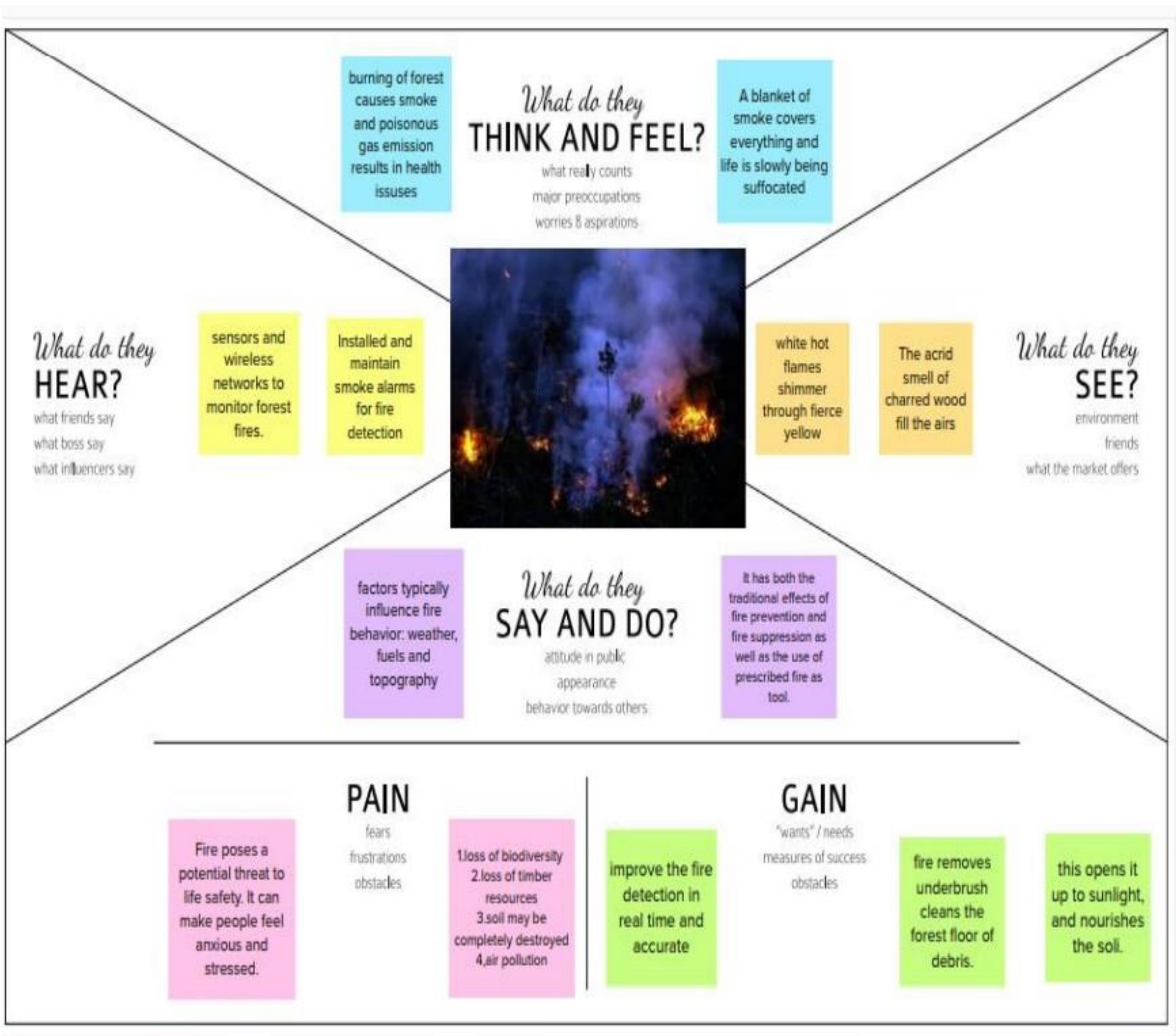

Fig 3.1

3.2 Ideation And Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Brainstorm, Idea Listing and Grouping

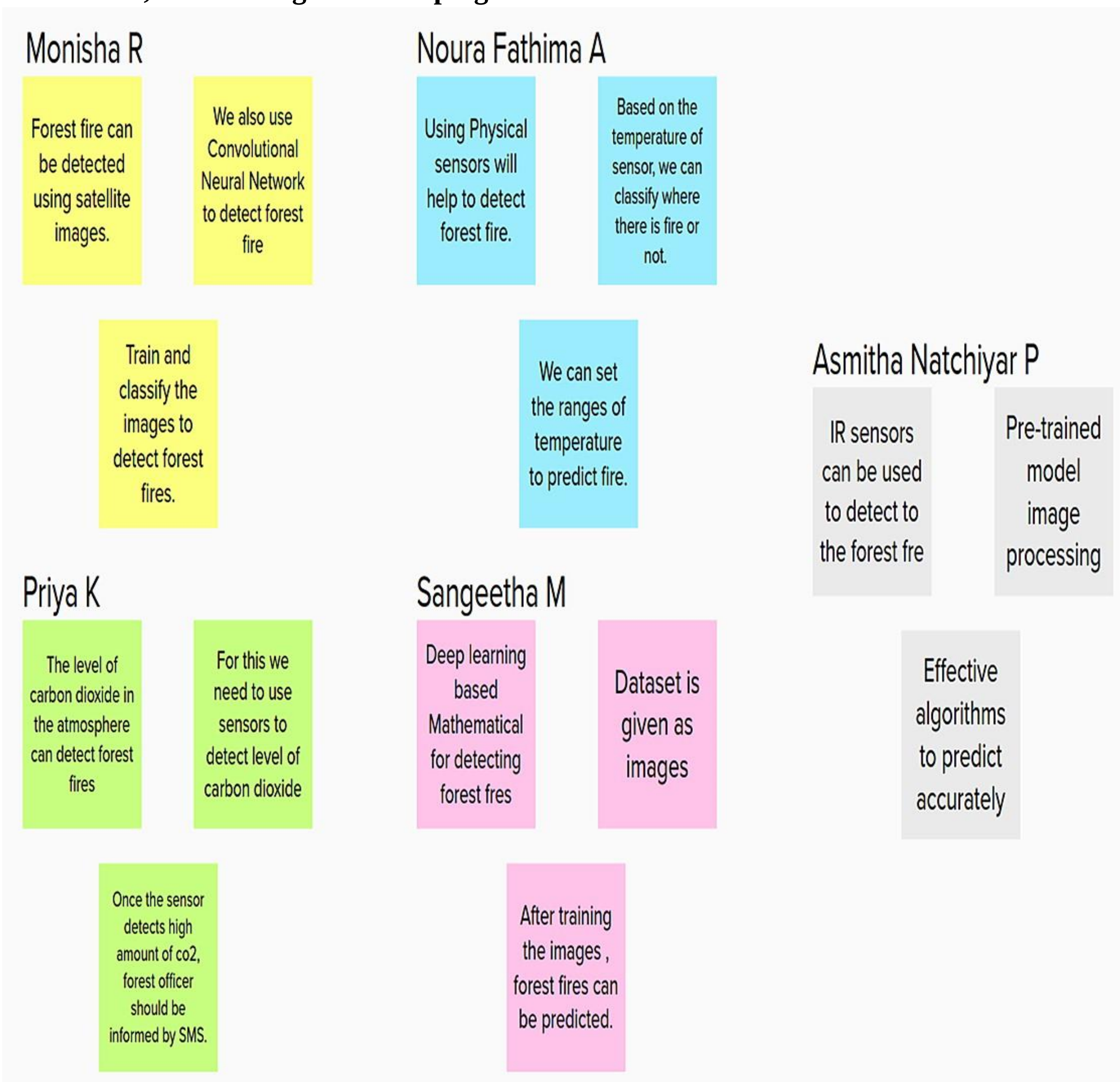

Fig 3.2

Idea Prioritization

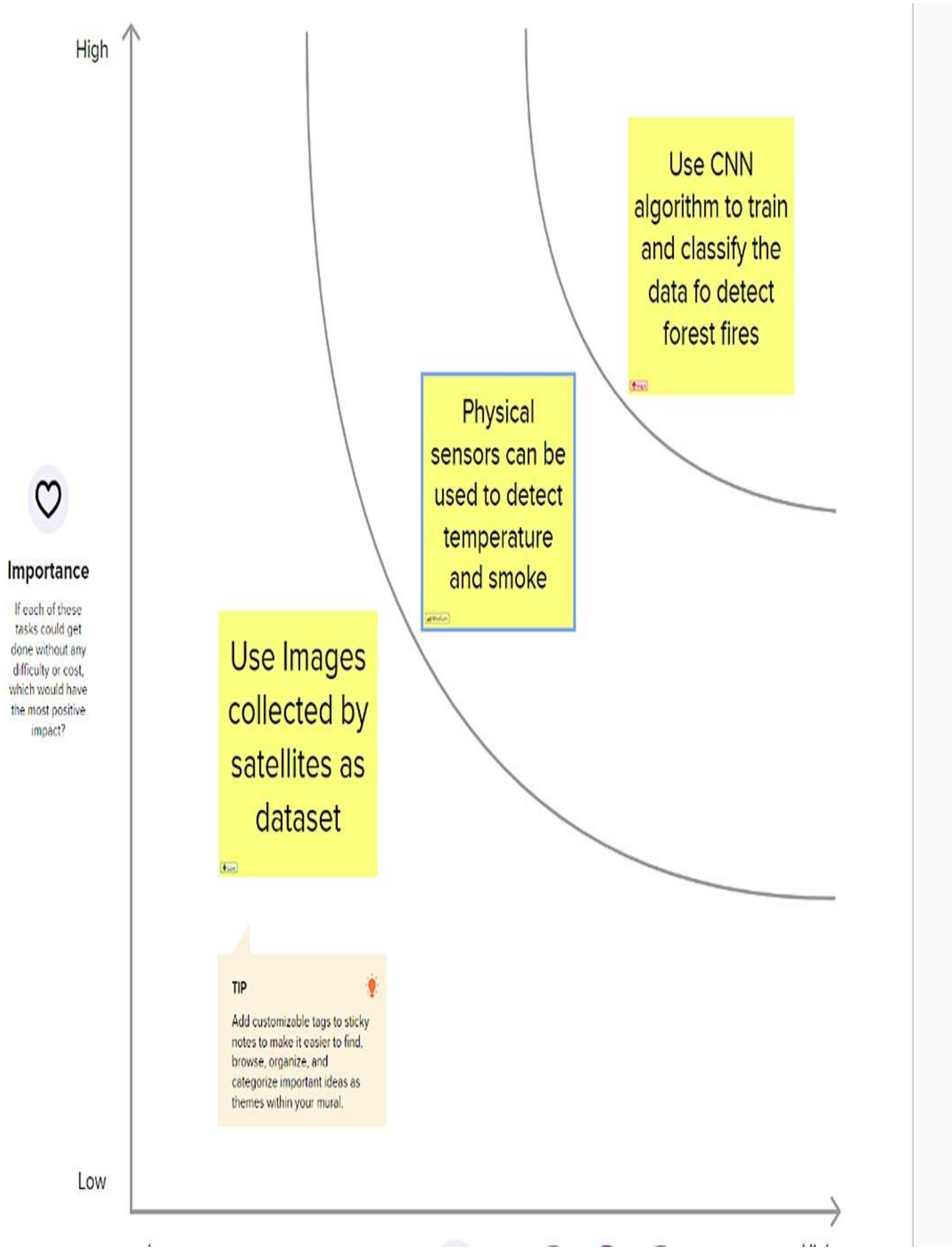

Fig 3.3

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Forest fires are a major environmental issue, creating economic and ecological damage while endangering human lives. It is difficult to predict and detect Forest Fire in a sparsely populated forest area So it is necessary to Detect forest fires in an early stage to avoid massive damage.
2.	Idea / Solution description	Identifying huge forest fires in real-time utilizing AI algorithms with camera and satellite footage. The systems then notify dispatchers and local authorities about the new ignition
3.	Novelty / Uniqueness	Convolutional Neural Network system allows us to deliver information more quickly and accurately. It is possible to deploy a comprehensive coverage, which is nearly impossible
4.	Social Impact / Customer Satisfaction	Monitoring of the potential danger regions and early identification of fire can greatly minimize the response time, as well as potential damage and frightening expenses, while also saving many lives.
5.	Business Model (Revenue Model)	The application is based on Subscription Model

6.	Scalability of the Solution	Can monitor different places simultaneously and can detect fire accurately. Despite the physical distance between resources and users, its regionally scalable system maintains its usability and utility;

3.4 Problem Solution Fit

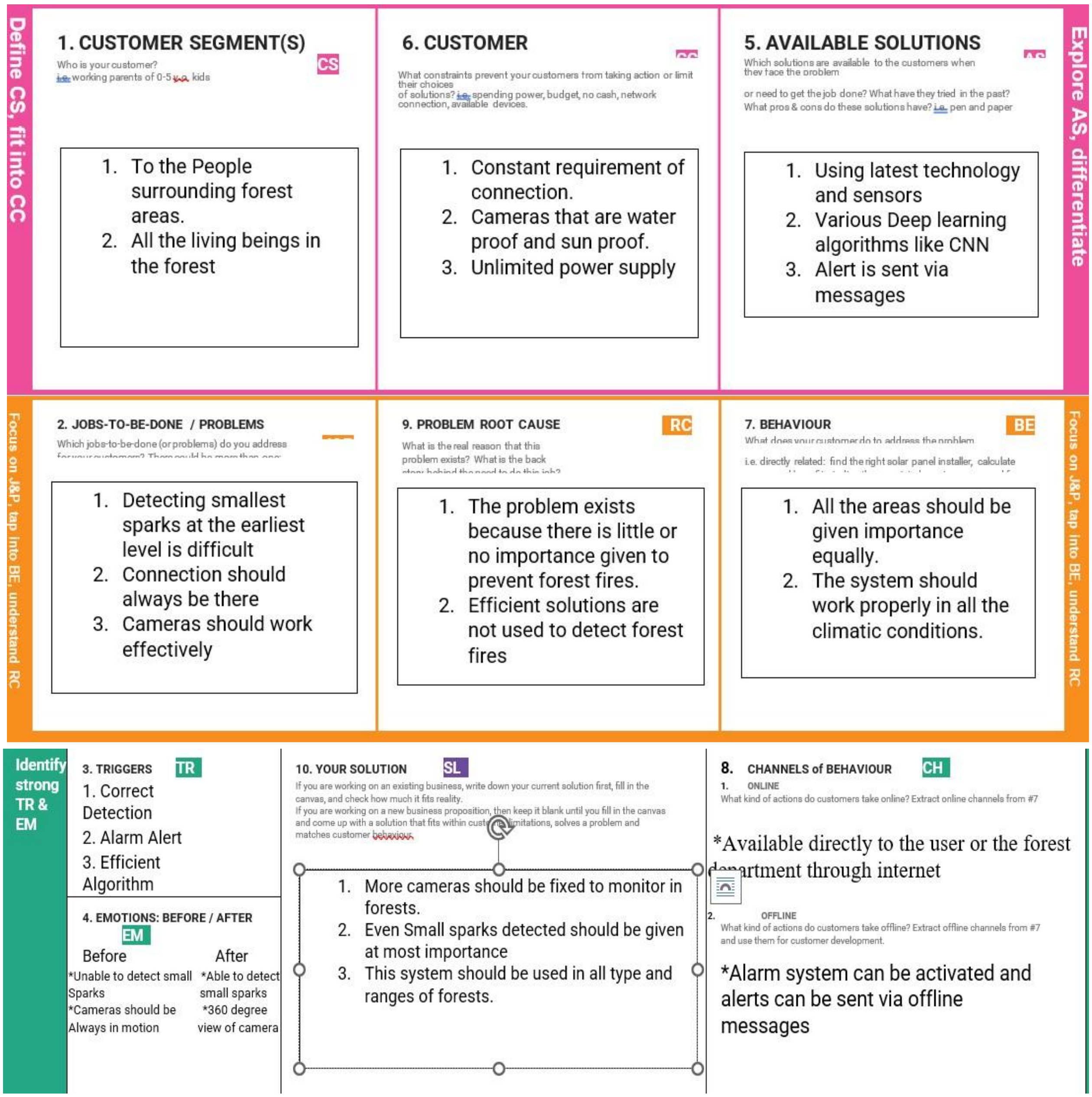

fig 3.4

Chapter 4

REQUIREMENT ANALYSIS

4.1 Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	Monitoring the smoke ,wind speed,CO2 and temperature using sensors	* Fire can be detected by using the amount of smoke. The smoke sensors are used to measure the amount of smoke from the fire, and it could be compared with a threshold value and if it is beyond that value. * The wind speed is calculated by the wind sensor nodes, which are manually placed in the forest. * When forest fires burn, they emit large volumes of carbon dioxide gas (CO2); you can use a network , CO2 and temperature sensors for forest fire detection.
FR-2	Unmanned aerial vehicle(UAV)	The forest fires can be detected by visionbased fire detection systems which can be mounted to an unmanned aerial vehicle (UAVs) for strategically scanning acreage of fire prone areas.

FR-3	Image processing by CCTV cameras	The cameras rotate continuously, scan the countryside with color, monochrome and near-infrared detectors. A feature-based AI algorithm uses an artificial neural network to scan the images for the telltale heat and smoke signature of wildfires. Under the right conditions, it can see as far as 40 miles away
FR-4	Data processing using Real time algorithm	The algorithm processes the data in real time on dedicated servers on site.It uses a cloud-based deep learning AI to detect and verify wildfire events in real time, drawing from satellite imagery and historical data.
FR-5	Light detection and ranging (LIDAR)	The system is used for the forest fire detection with the help of neural network .LIDAR is mainly used in the environmental and atmospheric studies. A lidar contains a photo detector, radiation emitter, signal receiver and signal processing hardware and software.
FR-6	Localization of fire	It use GPS to **track their location** as they can sends these location details along with the data such as measurements of temperature to cloud (or) other cloud based server.

4.2 Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	AI devices with machine learning verifies that usability is a special and important perspective to analyze user requirements, which can further improve the design quality.
NFR-2	**Security**	* Powerful CCTV and HD cameras are used. * Monitors 24/7 * IR flame detectors are used * Avoid intentional acts of arson
NFR-3	**Reliability**	A real-time and reliable fire detection method for an early warning system is required so that an immediate response to an incident can be made effective.

NFR-4	**Performance**	The system is designed for monitor the causing factors of forest fires such as temperature, humidity , air pressure level,oxygen and Carbon dioxide on the surface of air by using sensors. The CCTV cameras is used for image processing and detecting the forest fire. The GPS is used to track the location of forest fire.
NFR-5	**Availability**	By developing more advanced system by integrating wireless sensors with CCTV for added protection and precision. The algorithm shows great promise in adapting to various environment.
NFR-6	**Scalability**	By detecting the forest fire we can reduce air pollution, landslides,soil erosion by protecting strong rooted trees,the emission of CO_2 into the air during fire,No loss of life and resources.

Chapter 5
PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

FLOW:

- It is difficult to predict and detect Forest Fire in a sparsely populated forest area.
- It is more difficult if the prediction is done using ground-based methods like Camera or Video-Based approach.
- Satellites can be an important source of data prior to and also during the Fire due to its reliability and efficiency.
- The various real-time forest fire detection and prediction approaches, with the goal of informing the local fire authorities.
- If the fire is not detected it will send the result to the frame camera. if the forest fire will detected the alert will go to the video feed frame camera.

DIAGRAM

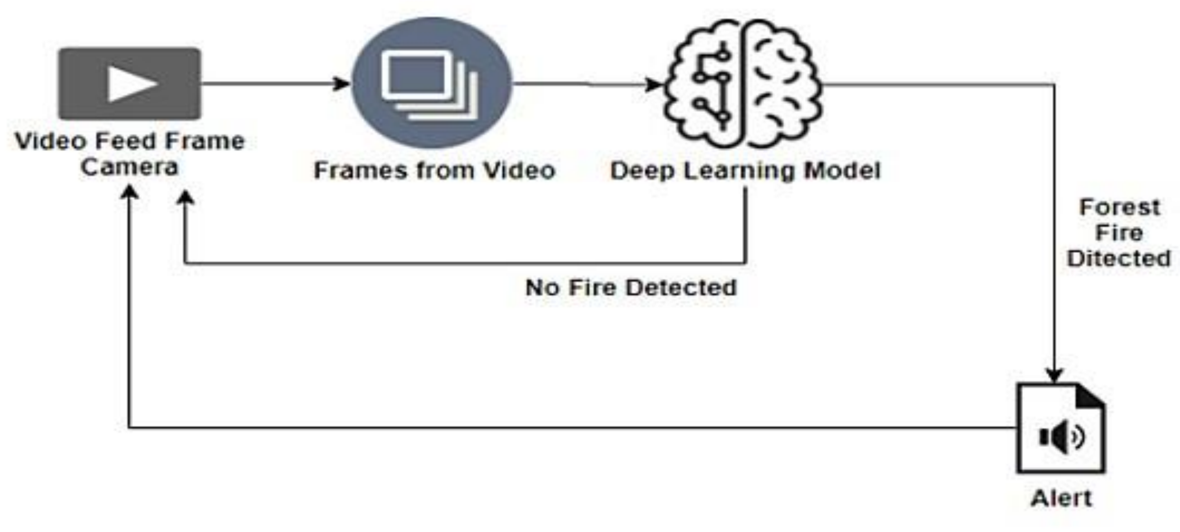

Fig 5.1

5.2 Solution & Technical Architecture:

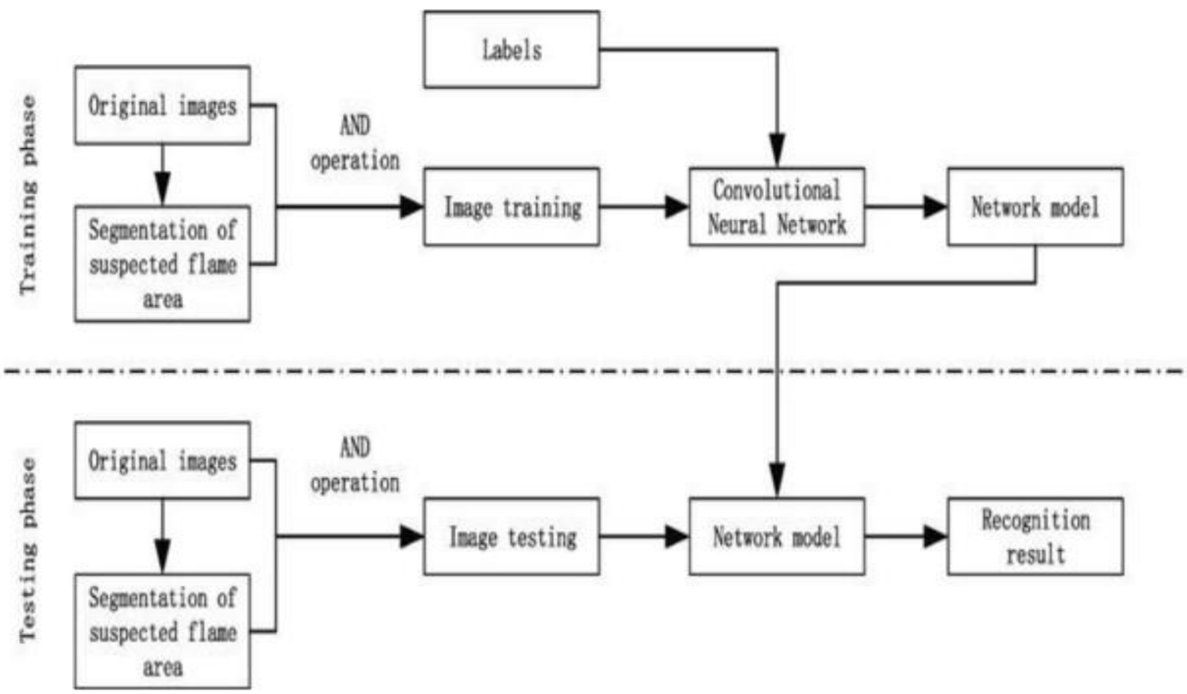

Fig5.2

Technical Architecture:

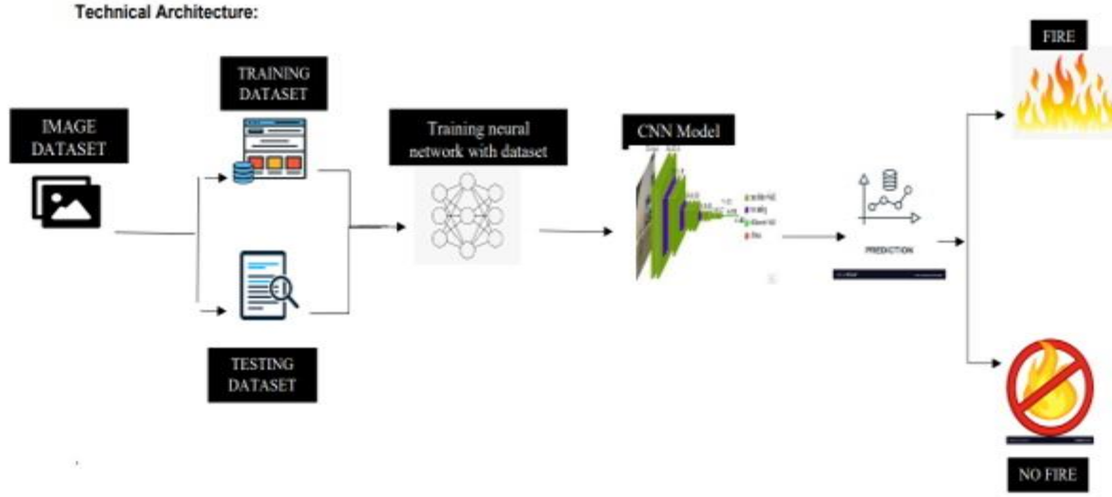

Fig 5.3

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story I Task	Acceptance criteria	Priority	Release
Environmentalist	Collect the data	USN-1	As an Environmentalist,it is necessary to collect the data of the forest which includes temperature,humidity,wind and rain of the forest	It is necessary to collect the right data else the prediction may become wrona	High	Sprint-1
		USN-2	Identify algorithms that can be used for prediction	To collect the algorithm to identify the accuracy level of each algorithms	Medium	Sprint-2
	Implement Algorithm	USN-3	Identify the accuracy of each algorithms	Accuracy of each algorithm-calculated so that it is easy to obtain the most accurate output	High	Sprint-2
		USN-4	Evaluate the Dataset	Data is evaluated before processing	Medium	Sprint-1
	Evaluate Accuracy of Algorithm	USN-5	Identify accuracy,precision,recall of each algorithms	These values are important for obtaining the riaht output	High	Sprint-3

Fig 5.4

Chapter 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Registration	USN-1	I can sign up for the application as a user by providing my email address, a password, and a password confirmation.	20	High
Sprint-1		USN-2	Once I've signed up to use the application, I, as a user,will receive a confirmation email.	20	Medium
Sprint-2	Input	USN-3	The database receives information each time a fire is discovered.	20	High
Sprint-2		USN-4	The warning system is turned on when there is a wildfire.	20	High
Sprint-3	Output	USN-5	Additionally, the alarm informed the relevant departments that a wildfire had started.	20	High
Sprint-4	Action	USN-6	The necessary steps will be taken to control the erupted wildfire by getting there as quickly as possible with the aid of detecting systems.	20	High

6.2 Sprint Delivery Schedule

Sprint	Total Story Poin ts	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Comple ted (as on Planne dEnd Date)	Sprint ReleaseDa te (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

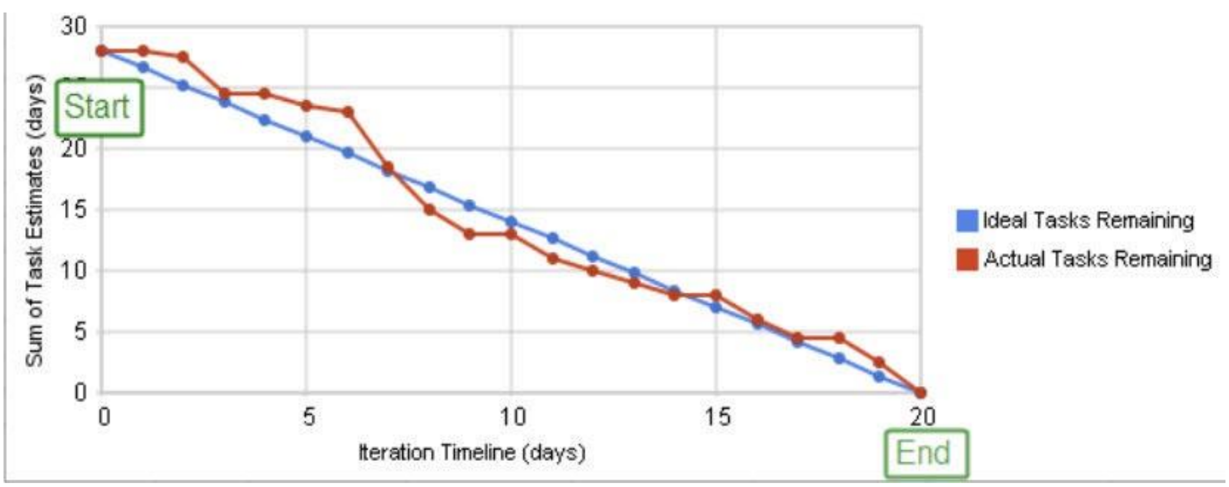

Fig 6.1

Chapter 7

CODING & SOLUTIONING

7.1 Feature 1

1. Preprocessing the dataset which consists of two classes of data(fire,no fire).

Image Preprocessing

#1.Importing the ImageDataGenerator Library

import numpy **as** np***7.***

import keras **from** sklearn.model_selection **import** train_test_split **from** keras.models **import** Sequential, load_model **from** keras.preprocessing.image **import** ImageDataGenerator **from** keras.callbacks **import** ModelCheckpoint, EarlyStopping, TensorBoard **from** keras.callbacks **import** ReduceLROnPlateau

from keras.layers **import** Conv2D, Dropout, Dense, Flatten, MaxPooling2D, SeparableConv2D, Activation, BatchNormalization **import** matplotlib.pyplot **as** plt **import** time **import** os

import tensorflow **as** tf

#2.Define parameters for ImageDataGenerator Class

```
train_datagen=ImageDataGenerator(rescale=1./255,
                                 shear_range=0.2,
                                 rotation_range=180,
                                 zoom_range=0.2,
                                 horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
```

#3.Applying ImageDataGenerator Functionality to Trainset and Testset

#*a. For Dataset* x_dataset =train_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/forest_fire",target_size = (128,128), class_mode = "binary",batch_size=32)

#*b. For Trainset*

x_train =train_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/fore st_fire/Training and Validation",target_size = (128,128), class_mode = "binary",batch_size=32) # *c. For Testset*

x_test =test_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/forest_fire/Testing",target_size = (128,128), class_mode = "binary", batch_size=32) x_train.class_indices

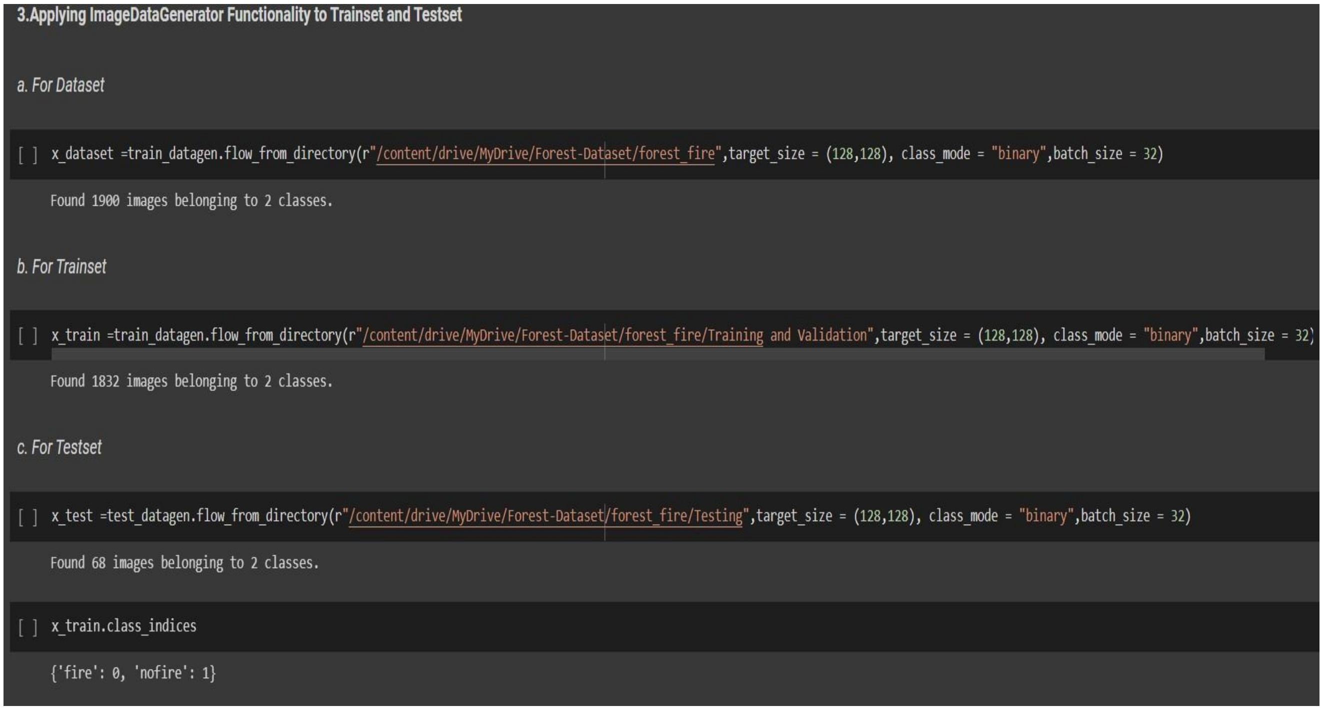

Fig 7.1

2.Building up a sequential model to train the dataset.

 # 1.Importing the Model Building Libraries

#Importing model libraries

from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense from tensorflow.keras.layers import Convolution2D from tensorflow.keras.layers import MaxPooling2D from tensorflow.keras.layers import Flatten

import warnings

warnings.filterwarnings('ignore')

 # 2.Initializing the Model

model=Sequential()

 # 3.Adding CNN Layers #

a. Adding Convolutional layer

model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))

b. Adding Pooling Layer

model.add(MaxPooling2D(pool_size=(2,2)))

c. Adding Flatten Layer

model.add(Flatten())

#Model Summary

model.summary()

```
[24] #Model Summary
     model.summary()

     Model: "sequential"
     _________________________________________________________________
      Layer (type)                Output Shape              Param #
     =================================================================
      conv2d (Conv2D)             (None, 126, 126, 32)      896

      max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
      )

      flatten (Flatten)           (None, 127008)            0

     =================================================================
     Total params: 896
     Trainable params: 896
     Non-trainable params: 0
     _________________________________________________________________
```

Fig 7.2

3. Prediction of data.

 # Predictions

pred = model.predict(x)

pred = np.round(pred)

pred

def predictImage(filename):

  img1=image.load_img(filename,target_size=(128,128))

  plt.imshow(img1)

  y=image.img_to_array(img1)

```
 x=np.expand_dims(y,axis=0)
 val=model.predict(x)
 print(val)
 if val==0:
   plt.xlabel(" No Fire",fontsize=30)
 elif val==1:
   plt.xlabel("Fire",fontsize=30)
predictImage("/content/drive/MyDrive/ForestDataset/fo
rest_fire/Testing/fire/abc173.jpg")
plt.xlabel("fire",fontsize=30)
predictImage('/content/drive/MyDrive/ForestDataset/for
est_fire/Testing/nofire/abc377.jpg')     plt.xlabel("     NO
fire",fontsize=30)
```

Fig 7.3

```
[128] predictImage('/content/drive/MyDrive/Forest-Dataset/forest_fire/Testing/nofire/abc377.jpg')
      plt.xlabel(" NO fire",fontsize=30)

      1/1 [==============================] - 0s 40ms/step
      [[0.48906803]]
      Text(0.5, 0, ' NO fire')
```

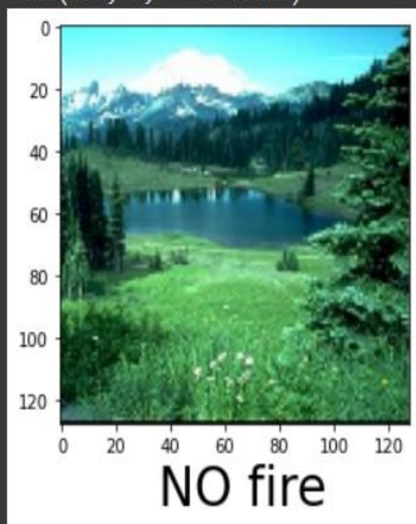

Fig 7.4

7.2 Feature 2

1.Creation of twilio account

To send an outgoing SMS message from your Twilio account you’ll need to make an HTTP POST to Twilio's Message resource.

Twilio's Python library helps you to create a new instance of the Message resource, specifying the To, From, and Body parameters of your message.

Replace the placeholder values for account_sid and auth_token with your unique values. You can find these in your Twilio console.

You’ll tell Twilio which phone number to use to send this message by replacing the from_ number with the Twilio phone number you purchased earlier.

Next, specify yourself as the message recipient by replacing the to number with your mobile phone number. Both of these parameters must use E.164 formatting (+ and a country code, e.g., +16175551212)

We also include the body parameter, which contains the content of the SMS we're going to send.

Twilio Account

```
account_sid =
'AC0919212286ee0fa7e4228913458e

auth_token =
'be43a0e16f576d34aad641b0cec546

client = Client(account_sid,
auth_token)

message = client.messages \

    .create(

         body='Forest
    fire is detected ,
    stay alert',

    from_='+16075363954',

    to='+919488200286'
     )

print(message.sid)
```

2. Sending SMS alert

pip install
twilio pip
install
playsound
#import opencv
librariy import
cv2 ***#import***
numpy import
numpy as np
#import image function from keras
from keras.preprocessing import

```
image #import load_model from
keras from keras.models import
load_model #import client from
twilio API from twilio.rest import
Client #imort playsound package
from playsound import playsound

#load the saved model model =
load_model(r'/content/drive/MyDrive/archive (1)/forest1.h5')
#define video
video = cv2.VideoCapture('/content/Fighting Fire with Fire _ Explained in 30
Seconds.mp4')
#define the features
name = ['forest','with
forest']

account_sid = ' AC0919212286ee0fa7e4228913458e863f'
auth_token = ' be43a0e16f576d34aad641b0cec546ef '
client = Client(account_sid, auth_token)

message = client.messages \
    .create(
         body='Forest fire is detected , stay alert',
         from_='+16075363954',
         to='+919488200286'
     )

print(message.sid)
```

Fig 7.5

Chapter 8

TESTING

8.1 Test Cases

Panel switches and keypads : TEST the operation of each control.

Visual indicators : TEST the operation of each visual indicator and alphanumeric displays.

Battery: MEASURE system quiescent and maximum alarm currents in accordance with Appendix . Calculate the required battery capacity and CHECK the nominal capacity of the installed batteries is not less than the calculated capacity.

Verify that the measured currents are the same as recorded in the baseline data.

8.2 User Acceptance Testing

1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the project at the time of the release to User Acceptance Testing (UAT).

2.Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	9	5	1	2	17
Duplicate	1	0	2	0	3
External	3	3	0	1	7
Fixed	10	2	3	20	35

Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	4	2	1	7
Totals	13	15	10	25	7 2

3.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	53	0	0	53
Security	2	0	0	2
Outsource Shipping	4	0	0	4
Exception Reporting	7	0	0	7
Final Report Output	3	0	0	3
Version Control	1	0	0	1

Chapter 9

RESULTS

9.1 Performance Metrics

1.Training the Model

Training the Model

```
] #fit or train the model
  r=model.fit_generator(x_train,steps_per_epoch=14,
                  epochs=10,validation_data=x_test,
                  validation_steps=2)
```

```
Epoch 1/10
14/14 [==============================] - 11s 704ms/step - loss: 3.6981 - accuracy: 0.7254 - val_loss: 1.6468 - val_accuracy: 0.8594
Epoch 2/10
14/14 [==============================] - 10s 719ms/step - loss: 0.6522 - accuracy: 0.8884 - val_loss: 0.6826 - val_accuracy: 0.8906
Epoch 3/10
14/14 [==============================] - 12s 827ms/step - loss: 0.2979 - accuracy: 0.9085 - val_loss: 0.3534 - val_accuracy: 0.9062
Epoch 4/10
14/14 [==============================] - 9s 655ms/step - loss: 0.2123 - accuracy: 0.9219 - val_loss: 0.2947 - val_accuracy: 0.9062
Epoch 5/10
14/14 [==============================] - 10s 664ms/step - loss: 0.2038 - accuracy: 0.9353 - val_loss: 0.2796 - val_accuracy: 0.9219
Epoch 6/10
14/14 [==============================] - 9s 652ms/step - loss: 0.2136 - accuracy: 0.9085 - val_loss: 0.1885 - val_accuracy: 0.9219
Epoch 7/10
14/14 [==============================] - 9s 656ms/step - loss: 0.2511 - accuracy: 0.9241 - val_loss: 0.4319 - val_accuracy: 0.8906
Epoch 8/10
14/14 [==============================] - 9s 602ms/step - loss: 0.2635 - accuracy: 0.9104 - val_loss: 0.5741 - val_accuracy: 0.8594
Epoch 9/10
14/14 [==============================] - 9s 650ms/step - loss: 0.3032 - accuracy: 0.8996 - val_loss: 0.4472 - val_accuracy: 0.8750
Epoch 10/10
14/14 [==============================] - 10s 699ms/step - loss: 0.2336 - accuracy: 0.9127 - val_loss: 0.3183 - val_accuracy: 0.8750
```

Fig 9.1

2.Loss or No loss

```
[100] import matplotlib.pyplot as plt
      plt.plot(r.history['loss'],label='loss')
      plt.plot(r.history['val_loss'],label='val_loss')
      plt.legend()

      <matplotlib.legend.Legend at 0x7f461beb8410>
```

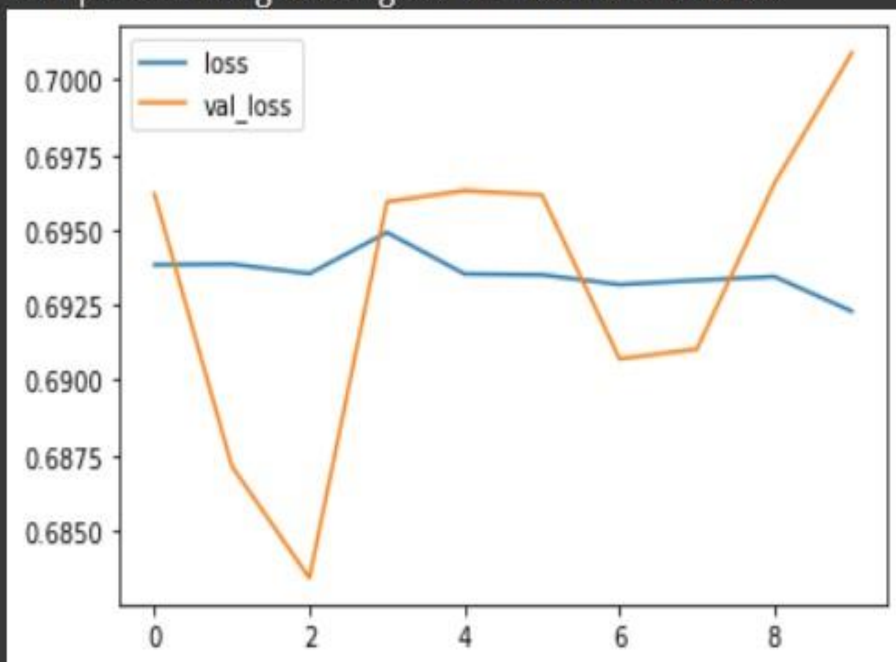

Fig 9.2

3.Accuracy Value

```
[101] plt.plot(r.history['accuracy'],label='acc')
      plt.plot(r.history['val_accuracy'],label='val_acc')
      plt.legend()

<matplotlib.legend.Legend at 0x7f461bec9d50>
```

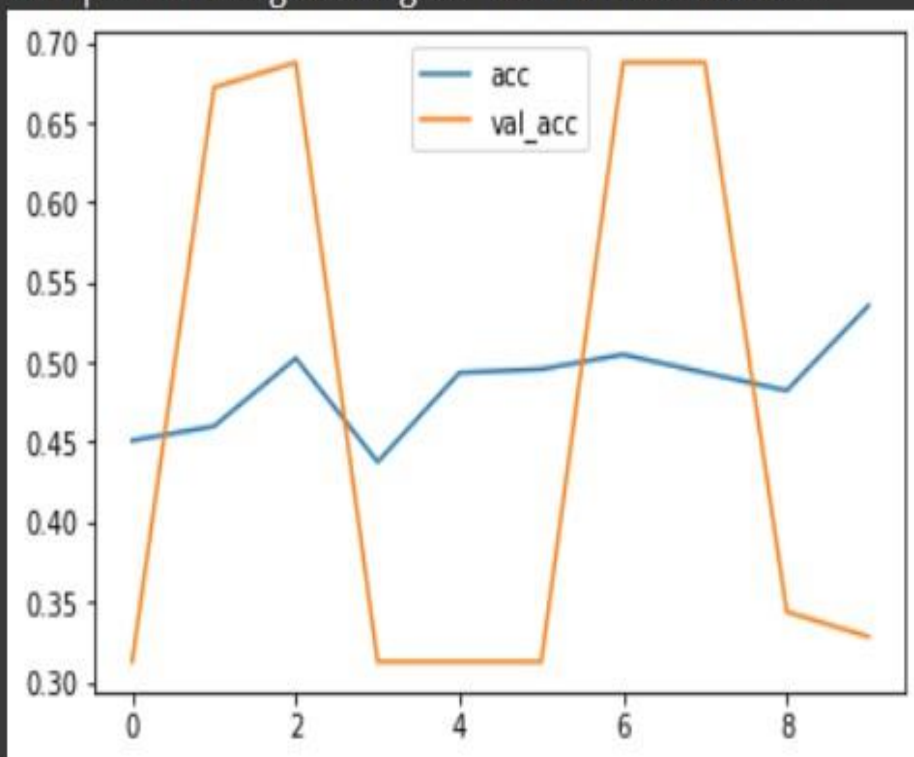

Fig 9.3

4.Confusion matrix

```
#confusion matrix
import matplotlib.pyplot as plt
import numpy
from sklearn import metrics

actual = numpy.random.binomial(1,.9,size = 1000)
predicted = numpy.random.binomial(1,.9,size = 1000)

confusion_matrix = metrics.confusion_matrix(actual, predicted)

cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])

cm_display.plot()
plt.show()
```

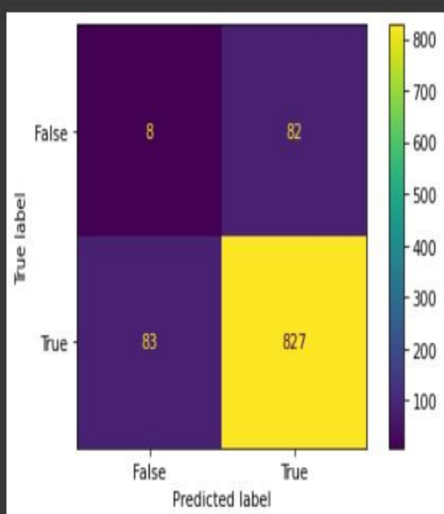

Fig 9.4

5.Predictions

```
[127] predictImage("/content/drive/MyDrive/Forest-Dataset/forest_fire/Testing/fire/abc173.jpg")
      plt.xlabel("fire",fontsize=30)
```

```
1/1 [==============================] - 0s 39ms/step
[[0.48906803]]
Text(0.5, 0, 'fire')
```

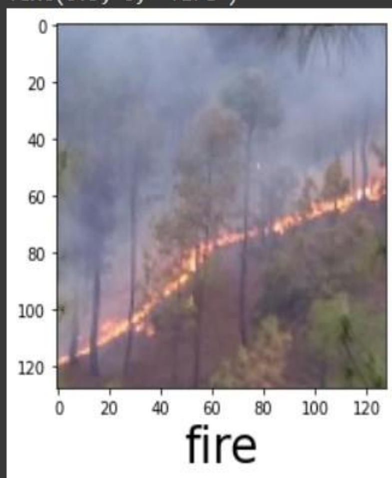

Fig 9.5

```
[128] predictImage('/content/drive/MyDrive/Forest-Dataset/forest_fire/Testing/nofire/abc377.jpg')
      plt.xlabel(" NO fire",fontsize=30)
```

```
1/1 [==============================] - 0s 40ms/step
[[0.48906803]]
Text(0.5, 0, ' NO fire')
```

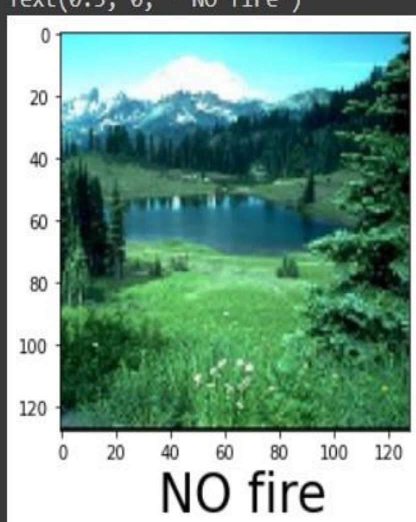

Fig 9.6

Chapter 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- The proposed system detects the forest fire at a faster rate compared to existing system. It has enhanced data collection feature.
- The major aspect is that it reduces false alarm and also has accuracy due to various sensors present.
- It minimizes the human effort as it works automatically.
- This is very affordable due to which can be easily accessed.
- The main objective of our project is to receive an alert message through an app to the respective user.
- The arrangement is fire-proof and can withstand high temperatures, rugged, reliable, cost-effective, and easy to install.
- It is also easy to decode the data from satellite at the ground station and no experts are required to understand or decode the data from the satellite.
- All the components like temperature sensor and the GPS are easy to interface.
- The approximate value of temperature and the GPS co-ordinates are obtained. Since we are using wireless sensing networks, the attenuation during the transmission of the signal or the data is minimised.
- It is More Reliable

DISADVANTAGES:

- The electrical interference diminishes the effectiveness of radio receiver.
- The main drawback is that it has less coverage range areas.
- Even a small fault would cause the whole system to fail.

Chapter 11
CONCLUSION

The proposed system for forest fire detection using wireless sensor networks and machine learning was found to be an effective method for fire detection in forests that provides more accurate results. Here, to obtain a more accurate outcome within the lowest latency, the analysis should take place continuously and camera monitoring should be effectively done. This system is well developed to fit any weather condition, climatic condition, or area, . In the case of node deployment, cameras can be mounted at any place in the forest even with good connectivity and built-in network infrastructure. IR frame sensors are used to enhance efficiency of the system. A unique feature that sends alert messages to the concerned authorities when fire is detected is also added. Thus, By detecting the forest fire we can reduce air pollution, landslides,soil erosion by protecting strong rooted trees,and the emission of CO2 into the air during fire causing no loss of life and resources.

Chapter 12
FUTURE SCOPE

- Right now we have designed the project for control of two devices but it can be designed for more numbers of devices.
- It can be further expanded with a voice interactive system facility.
- A feedback system can also be included which provides the state of device to the remote users.

Chapter 13

APPENDIX

Source Code

#Download the Dataset

pwd

#Load the Image Dataset
from google.colab import drive
drive.mount('/content/drive')

 # call load_data with allow_pickle implicitly set to true
import numpy as np

data = np.load('/content/drive/My Drive/Forest-Dataset/Dataset.zip', allow_pickle=**True**)
print('data loaded') cd //content/drive/MyDrive/Forest-Dataset

#Unzip the Dataset

!unzip Dataset.zip

#Image Preprocessing

#1.Importing the ImageDataGenerator Library **import** numpy **as** np **import** keras **from** sklearn.model_selection **import** train_test_split **from** keras.models **import** Sequential, load_model **from** keras.preprocessing.image **import** ImageDataGenerator **from** keras.callbacks **import** ModelCheckpoint, EarlyStopping, TensorBoard **from** keras.callbacks **import** ReduceLROnPlateau

from keras.layers **import** Conv2D, Dropout, Dense, Flatten, MaxPooling2D, SeparableConv2D, Activation, BatchNormalization **import** matplotlib.pyplot **as** plt **import** time **import** os

import tensorflow **as** tf

#2.Define parameters for ImageDataGenerator Class

train_datagen=ImageDataGenerator(rescale=1./255,

shear_range=0.2,

rotation_range=180,

zoom_range=0.2,

horizontal_flip=**True**)

test_datagen=ImageDataGenerator(rescale=1./255)

#3.Applying ImageDataGenerator Functionality to Trainset and Testset

#*a. For Dataset*

x_dataset =train_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/forest_fire",target_size = (128,128), class_mode = "binary",batch_size=32)

#*b. For Trainset*

x_train =train_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/forest_fire/Training and Validation",target_size = (128,128), class_mode = "binary",batch_size=32) # *c. For Testset*

x_test =test_datagen.flow_from_directory(r"/content/drive/MyDrive/ForestDataset/forest_fire/Testing",target_size = (128,128), class_mode = "binary", batch_size=32) x_train.class_indices ***# Model Building***

 # 1.Importing the Model Building Libraries

#Importing model libraries

from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense from tensorflow.keras.layers import Convolution2D from tensorflow.keras.layers import MaxPooling2D from tensorflow.keras.layers import Flatten

import warnings

warnings.filterwarnings('ignore') ***# 2.Initializing the Model***

model=Sequential()

 # 3.Adding CNN Layers # a. Adding Convolutional layer

model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
# b. Adding Pooling Layer
model.add(MaxPooling2D(pool_size=(2,2)))
# c. Adding Flatten Layer
model.add(Flatten( )) #Model Summary
model.summary() ***# 4.Adding Dense Layers***
#a. Adding Hidden layers #model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu')) #b. Adding Output layer
model.add(Dense(1,activation='sigmoid ')) ***# 5.Configuring the Learning Process***
model.compile(loss='binary_crossentropy',
optimizer='adam',
metrics=['accuracy'])
 # 6.Training the Model #fit or train the model
r=model.fit_generator(x_train,steps_per_epoch=14,
epochs=10,validation_data=x_test,
validation_steps=2)
#plotting loss value import matplotlib.pyplot as plt
plt.plot(r.history['loss'],label='loss')
plt.plot(r.history['val_loss'],label='val _loss')
plt.legend() #plotting accuracy value plt.plot(r.history['accuracy'],label='acc')
plt.plot(r.history['val_accuracy'],label='val _acc') plt.legend() ***# 7.Save the Model***
model.save("forest1.h5") ls forest_fire/

 # 8.Test The Model

import numpy as np

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import cv2

#load the model model=load_model('forest1.h5')
img=image.load_img('/content/drive/MyDrive/For
estDataset/forest_fire/Testing/fire/abc169.jpg') img
img=image.load_img('/content/drive/MyDrive/For
est-
Dataset/forest_fire/Testing/fire/abc169.jpg',target_size=(128,128))
img
x=image.img_to_array(img) x
x=np.expand_dims(x,axis=0) x
y=np.argmax(model.predict(x),axis=1) y
x_train.class_indices index=['fire','nofire']
index[y[0]] #Testing image 1
img=image.load_img('/content/drive/MyDrive/For
est-
Dataset/forest_fire/Testing/fire/abc183.jpg',target_size=(128,128))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
index=['fire','nofire']
print('Fire')
img

#Testing image 2

img=image.load_img('/content/drive/MyDrive/Forest-
Dataset/forest_fire/Testing/nofire/abc337.jpg',target_size=(128,128)
) x=image.img_to_array(img) x=np.expand_dims(x,axis=0)
index=['fire','nofire']
print('No
Fire') img

#Testing image 3

img=image.load_img('/content/drive/MyDrive/Forest-
Dataset/forest_fire/Testing/nofire/abc377.jpg',target_size=(128,128))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
index=['fire','nofire']

```
print('nofir
e') img
#Testing image 4
img=image.load_img('/content/drive/MyDrive/Forest-
Dataset/forest_fire/Testing/fire/abc173.jpg',target_size=(128,128))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
index=['fire','nofire']
print('fire')
img #
```

9.Predictions

```
pred =
model.predict(x)
pred =
np.round(pred)
pred
def predictImage(filename):
  img1=image.load_img(filename,target_size=(128,128))
  plt.imshow(img1)
  y=image.img_to_array(img1)
  x=np.expand_dims(y,axis=0)
  val=model.predict(x)

print
(val)
if
val=
=0:
    plt.xlabel(" No Fire",fontsize=30)
  elif val==1:
    plt.xlabel("Fire",fontsize=30)
predictImage("/content/drive/MyDrive/ForestDataset/forest_fire/Testing
/fire/abc173.jpg") plt.xlabel("fire",fontsize=30)
predictImage('/content/drive/MyDrive/ForestDataset/forest_fire/Testing/
nofire/abc377.jpg')
plt.xlabel(" NO fire",fontsize=30)
```