Emerging Methods for Early Detection of Forest

A Project report submitted in partial fulfilment of 7th semester in degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

TEAM ID	: PNT2022TMID40613
CHINNADURAI.I	: (610919104018)
VISHANTH.R	: (610919104103)
ARUNKUMAR.M	: (610919104015)
KADHARALI.S	: (610919104015)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
JAYALAKSHMI INSTITUTE OF TECHNOLOGY, THOPPUR

ANNA UNIVERSITY: CHENNAI 600025

NOV-2022

JAYALAKSHMI INSTITUTE OF TECHNOLOGY, THOPPUR
(A Constituent College of Anna University, Chennai)

<u>BONAFIDE CERTIFICATE</u>

Certified that this project report **"Emerging Methods for Early Detection of Forest Fires"** is the bonafide work done **CHINNADURAI.I (610919104018), VISHANTH.R (610919104103) , ARUNKUMAR.M(610919104015)** for **Nalayathiran** in **VII** semester of **B.E.,** degree course in **Computer Science and Engineering** branch during the academic year of **2022-2023.**

Staff-In charge
S.Subha.ME

Evaluator
G.GEETHA.ME

Head of the Department
MR SATHYA.ME

ACKNOWLEDGEMENT

We express our breathless thanks to our **Dr.VENKATESAN.ME., Ph.D.,** the PRINCIPAL(i\c), JAYALAKSHMI INSTITUTE OF TECHNOLOGY, THOPPUR, for giving constant motivation in succeeding in our goal.

We acknowledge our sincere thanks to Head of the Department (i/c)

MR SATHYA.ME for giving us valuable suggestion and help towards us throughout this Project.

We are highly grateful to thank our Project coordinator **S Subha.ME.,** and our Project Evaluator **G GEETHA M.E.,** Department of Computer Science and Engineering, JAYALAKSHMI INSTITUTE OF TECHNOLOGY , THOPPUR, for the coordinating us throughout this Project.

We are very much indebted to thank all the faculty members of Department of Computer science and Engineering in our Institute, for their excellent moral support and suggestions to complete our Project work successfully.

Finally our acknowledgment does our parents, sisters and friends those who had extended their excellent support and ideas to make our Project a pledge one.

CHINNADURAI I
VISHANTH R
ARUNKUMAR M
KADHARALI S

Abstract

A wildland fire is an uncontrolled fire that occurs mainly in forest areas, although it can also invade urban or agricultural areas. Among the main causes of wildfires, human factors, either intentional or accidental, are the most usual ones. The number and impact of forest fires are expected to grow as a consequence of the global warming. In order to fight against these disasters, it is necessary to adopt a comprehensive, multifaceted approach that enables a continuous situational awareness and instant responsiveness. This paper describes a hierarchical wireless sensor network aimed at early fire detection in risky areas, integrated with the fire fighting command centres, geographical information systems, and fire simulators. This configuration has been successfully tested in two fire simulations involving all the key players in fire fighting operations: fire brigades, communication systems, and aerial, coordination, and land means.

TABLE OF FIGURS

1. INTRODUCTION

Forest fires are a major environmental issue, creating economic and ecological damage while endangering human lives. There are typically about 100,000 wildfires in the United States every year. Over 9 million acres of land have been destroyed due to treacherous wildfires. It is difficult to predict and detect Forest Fire in a sparsely populated forest area and 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

1.1 Project Overview

The idea is to create aand develop a system that can identify the effects of the forest fire and it can analyse the forest fire by advanced AI techniques and CNN Algorithm then the Prediction model is Checked and then the model is connected with Twilio account credentials of the Developer consisting of phone numbers of the persons in the surroundings of the people in the area of easy forest fire zone then an security sound alert system is developed to make a alert sound which is downloaded from internet then the entire model is deployed to the IBM Cloud account that we have created.

1.2 Purpose

The forest fires destroys the wildlife habitat, damages the environment, affects the climate, spoils the biological properties of the soil, etc. So the forest fire detection is a major issue in the present decade. At the same time the forest fire have to be detected as fast as possible.

2. LITERATURE SURVEY

2.1 Existing problem

Forest fires have been and still are serious problem for the European Union and for all other countries in Europe. In the year 2000, the EU has established the European Forest Fire Information system (EFFIS) , which will soon become part of the European Emergency Management Service, maintained by the Copernicus Earth Observation Programme . This system provides valuable near real-time and also historical data on the forest fires in Europe, the Middle East and North Africa. Currently EFFIS is being used and supported with data by 25 EU member states and by numerous other countries. According to the annual report of EFFIS for 2016 , more than 54 000 forest fires have occurred all around Europe and they have led to nearly 376 thousand hectares of burnt areas .If we compare these values to the average values from the EFFIS reports for the period 2006-2015, the number of forest fires have decreased by 13327 or by nearly 20%. This decrease can be explained with the more severe actions and sanctions towards the arsonists and with the introduction of more advanced technical solutions for early detection of the fires. Even though their number is decreasing, the forest fires continue to be extremely devastating events and they have destroyed just 27 thousand hectares (or 6.6 %) less than the average burnt areas for the period 2006-2015, according to . Confirmation for this are the devastating forest fires form 2018, which took place in the Attica region of Greece and led to more than 90 fatalities and to more than 200 injured people,as well as to the destruction to thousands of buildings . Forest Fires can be divided into 4 categories in the forests of Hungary based on tree and other vegetation species: • underground burning, peat fire; • fire in undergrowth or dead fallen leaves; • fire in seedlings and saplings; • fire in trunks and shrouds.

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_i
n_Europe_Middle_east_and_North_Africa_2016_final_pdf_JZU7He L.pdf
[4] The 2018 Attica wildfires Wikipedia webpage available at
https://en.wikipedia.org/wiki/2018_Attica_wildfires
[5] Rajmund Kuti,"Characterstic of forest fire and its impact on environment",(2016)

2.3 Problem Statement Definition

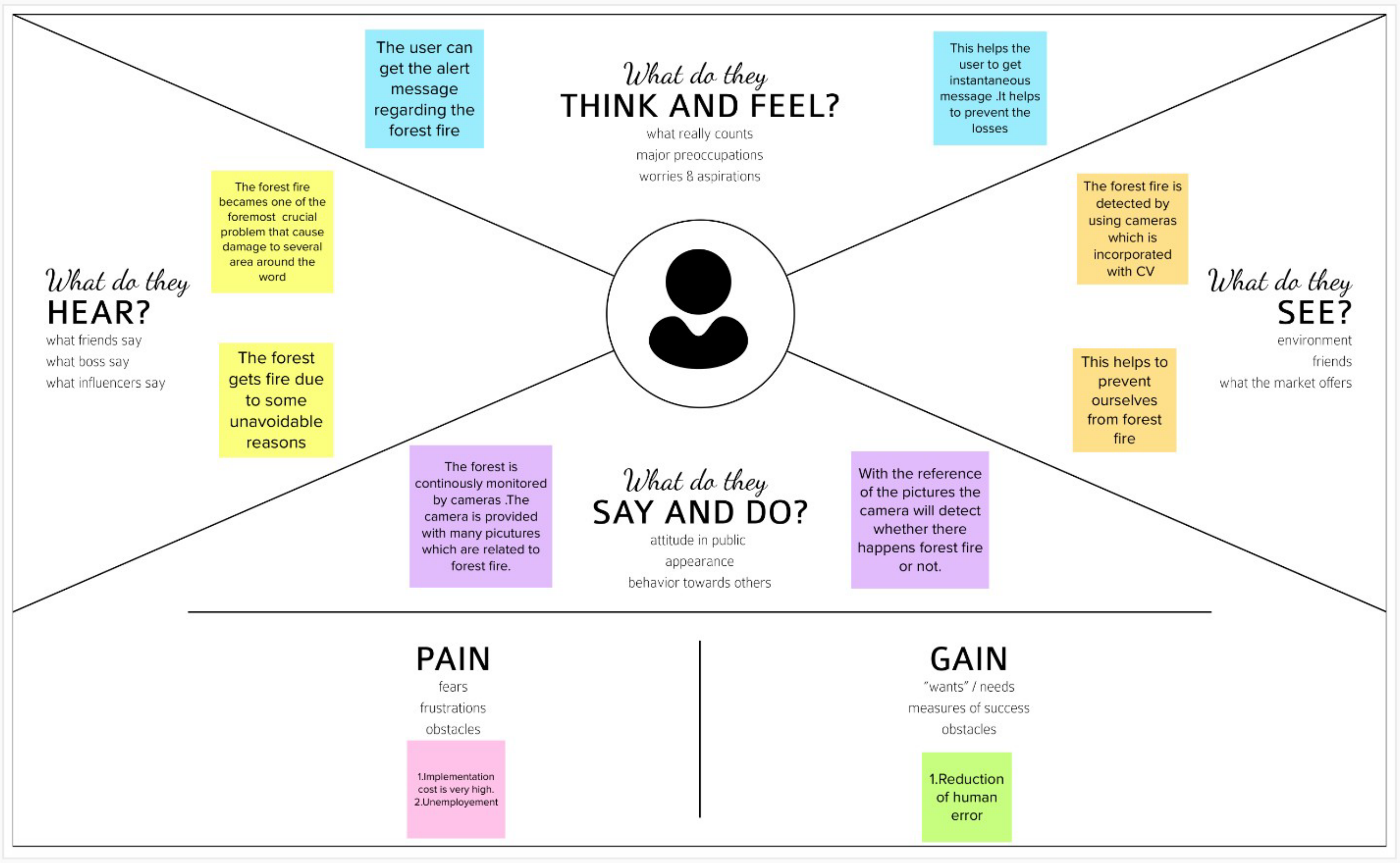
The user interacts with a web camera to read the video.
Once the input image from the video frame is sent to the model, if the fire is detected it is showcased on the console, and alerting sound will be generated and an alert message will be sent to the Authorities.

- Data Collection.
  - Collect the dataset or create the dataset.
- Image Preprocessing.
  - Import ImageDataGenerator Library.
  - Define the parameters /arguments for ImageDataGenerator class
  - Applying ImageDataGenerator on trainset and test set.
- Model Building
  - Import the model building Libraries
  - Initializing the model
  - Adding CNN Layers
  - Adding Hidden Layer
  - Adding Output Layer
  - Configure the Learning Process
  - Training and testing the model
  - Optimize the Model
  - Save the Model
- Video Streaming and alerting
  - OpenCV for video processing
  - Creating an account in Twilio service
  - Use Twilio API to send messages.

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map canvas is a more in-depth version of the original empathy map, which helps identify and describe the user's needs and pain points.



The user can get the alert message regarding the forest fire

This helps the user to get instantaneous message .It helps to prevent the losses

The forest fire becames one of the foremost crucial problem that cause damage to several area around the word

The forest gets fire due to some unavoidable reasons

The forest fire is detected by using cameras which is incorporated with CV

This helps to prevent ourselves from forest fire

The forest is continously monitored by cameras .The camera is provided with many picutures which are related to forest fire.

With the reference of the pictures the camera will detect whether there happens forest fire or not.

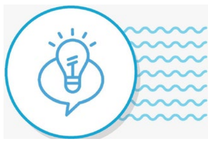
1.Implementation cost is very high.
2.Unemployement

1.Reduction of human error

3.2 Ideation & Brainstorming

organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

(9 10 minutes to prepare

J;/ 1 hour to collaborate

2-8 people recommended

Share template feedback

0

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

(!) 10 minutes

0 Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead

0 Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

0 Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session

Open article

0

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

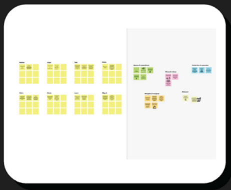
(!) 5 minutes

How might we detect the Forest Fires Early to prevent the loss of valuable timber resources?

Key rules of brainstorming

To run an smooth and productive session

8	Stay in topic	-$-	Encourage wild ideas.
8	Defer judgment		Listen to others
	Go for volume	®	If possible. be visual.

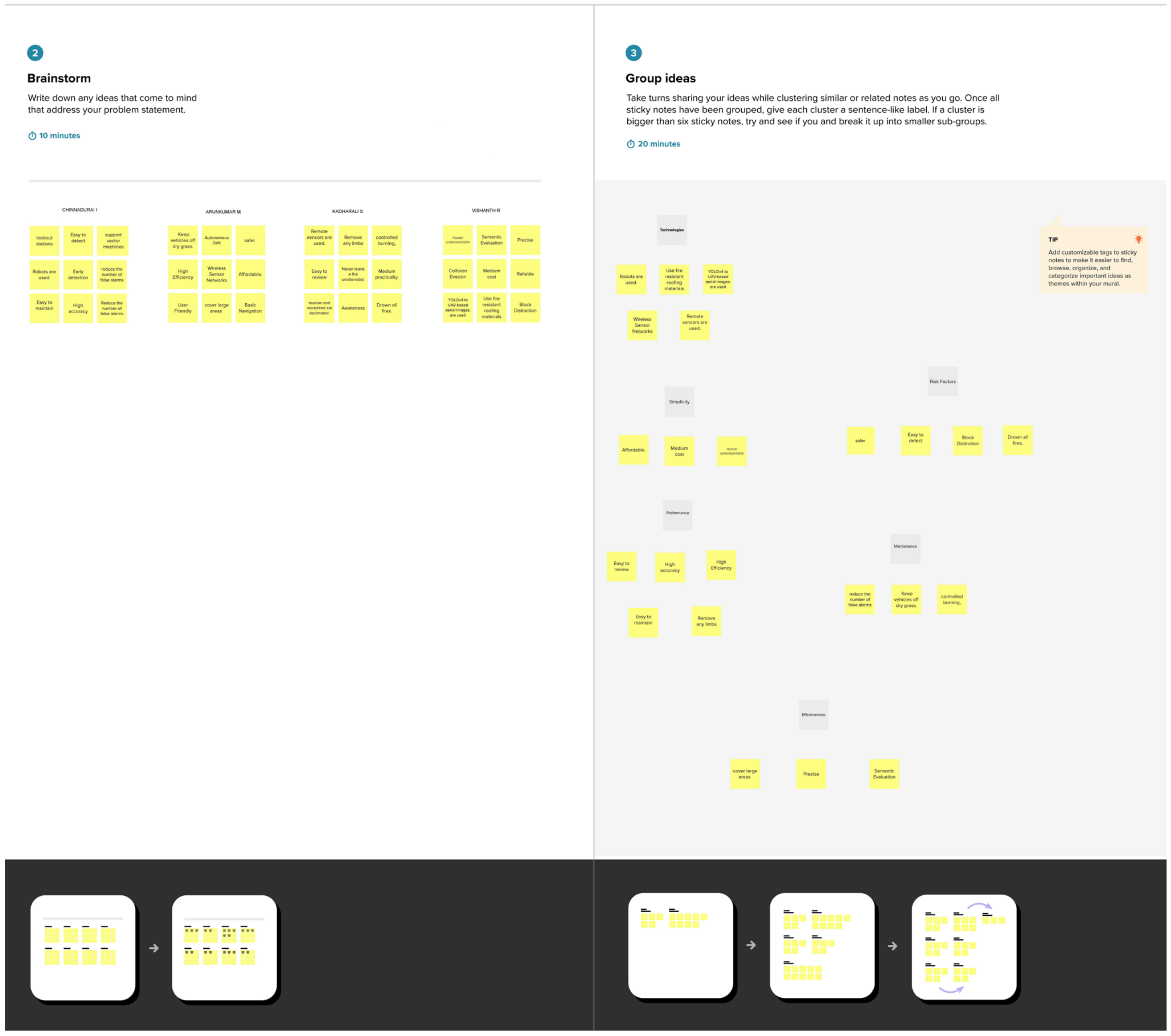


Need some inspiration?

See a finished version of this template to kickstart your work.

Open example →

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

CHINNADURAI I

lookout stations
Easy to detect
support vector machines
Robots are used.
Early detection
reduce the number of false alarms
Easy to maintain
High accuracy
Reduce the number of false alarms

ARUNKUMAR M

Keep vehicles off dry grass.
Autonomous Drift
safer
High Efficiency
Wireless Sensor Networks
Affordable.
User Friendly
cover large areas
Basic Navigation

KADHARALI S

Remote sensors are used.
Remove any limbs
controlled burning,
Easy to review
Never leave a fire unattended.
Medium practicality.
tourism and recreation are decimated.
Awareness
Drown all fires.

VISHANTH R

Human understandable
Semantic Evaluation
Precise
Collision Evasion
Medium cost
Reliable
YOLOv4 to UAV-based aerial images. are used
Use fire resistant roofing materials
Block Distinction

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Technologies

Robots are used.
Use fire resistant roofing materials
YOLOv4 to UAV-based aerial images. are used
Wireless Sensor Networks
Remote sensors are used.

Simplicity

Affordable.
Medium cost
Human understandable

Risk Factors

safer
Easy to detect
Block Distinction
Drown all fires.

Performance

Easy to review
High accuracy
High Efficiency
Easy to maintain
Remove any limbs

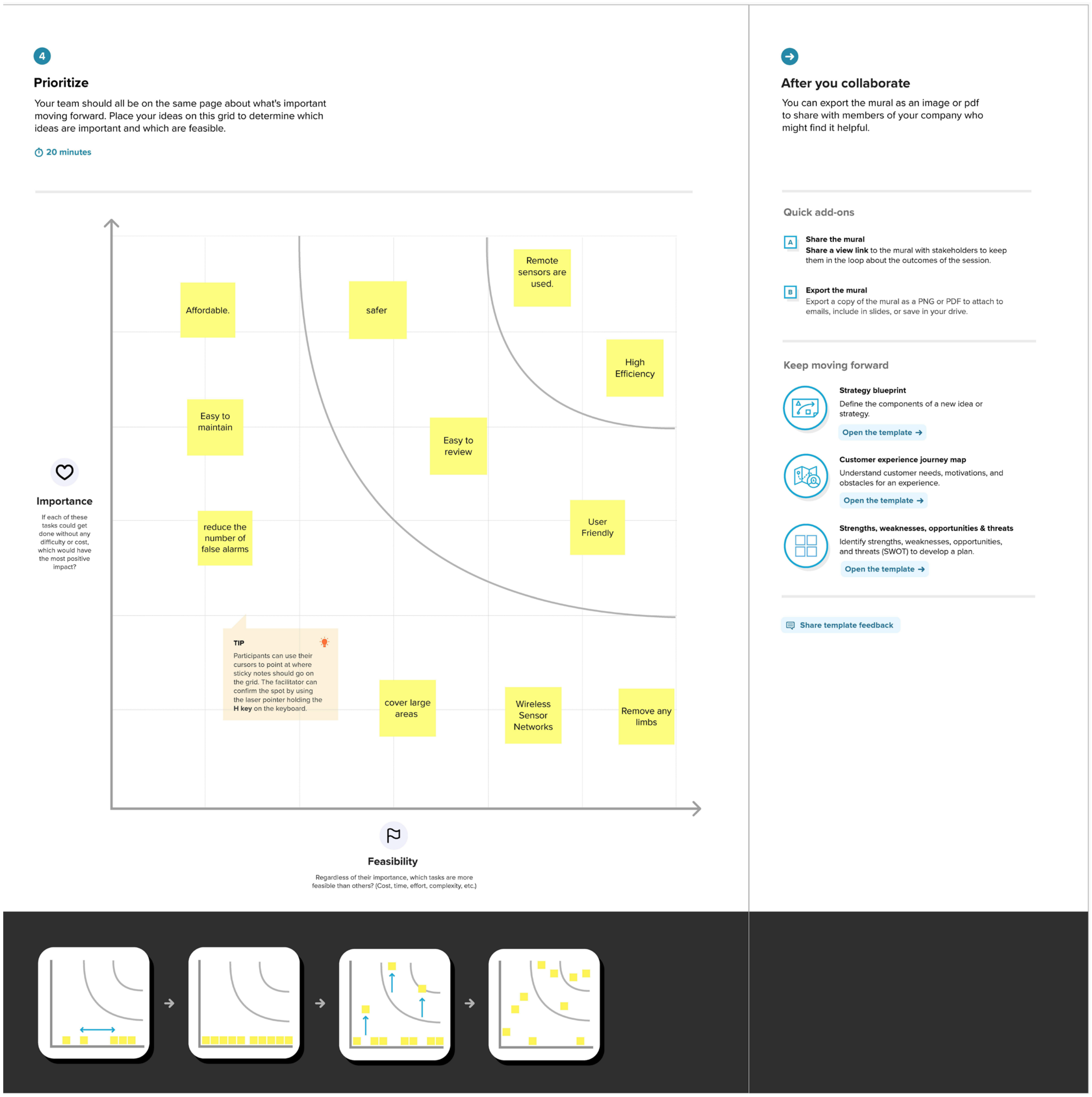
Maintanence

reduce the number of false alarms
Keep vehicles off dry grass.
controlled burning,

Effectiveness

cover large areas
Precise
Semantic Evaluation

Step-3: Idea Prioritization



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

3.3 Proposed Solution:

S.No.	**Parameter**	**Description**
1.	Problem Statement (Problem to be solved)	Emerging methods for early detection of forests fires
2.	Idea / Solution description	Forest surveillance video cameras can be used to monitor the forest areas and they can alert the forest department if there is any symptoms of forest fire or any other suspicious activities.
3.	Novelty / Uniqueness	The digital image processing technique, pattern-recognition technology and reinforcement learning can greatly improve the sensing of forest fire and they are much more effective as they improve forecast and reaction time is much less
4.	Social Impact / Customer Satisfaction	This product has huge social and biological impact as prevention of forest fire can save countless acres of forest land and wild lives. Forest fire also increases the amount of CO2 in the atmosphere. So prevention of forest fire can also reduce global warming.
5.	Business Model (Revenue Model)	This product can be only used by a giant corporation or a government to monitor huge reserve forests. This can be considered as a profitable and useful product as government spends millions of dollars for detection of forest fires and millions more if there is any actual forest fire like rescuing and stopping the fire.
6.	Scalability of the Solution	It is highly challenging to implement this method as we have to install hundreds of cameras to cover a respectable amount of area. The cameras need to be connected to electricity and they also need to be connected to internet to process the image and analyze them. They need to be connected to local server which should be located at the middle of the forest.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

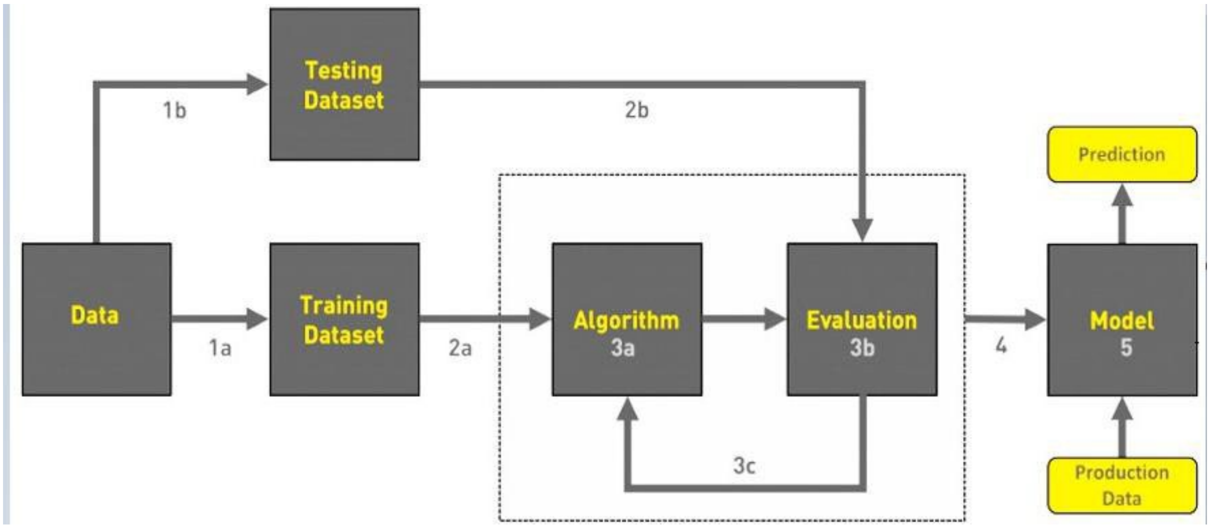
S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Python Flask framework is used	Technology of Opensource framework
2.	Security Implementations	Mandatory Access Control (MAC) and Preventative Security Control is used	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	High scalability with 3-tier architecture	Web server – HTML ,CSS ,JavaScript Application server – Python , Anaconda Database server –IBM DB2
4.	Availability	Use of load balancing to distribute traffic across servers	IBM load balancer
5.	Performance	Enhance the performance by using IBM CDN	IBM Content Delivery Network

4.2 Non-Functional requirements

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Python Flask framework is used	Technology of Opensource framework
2.	Security Implementations	Mandatory Access Control (MAC) and Preventative Security Control is used	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	High scalability with 3-tier architecture	Web server – HTML ,CSS ,JavaScript Application server – Python , Anaconda Database server –IBM DB2
4.	Availability	Use of load balancing to distribute traffic across servers	IBM load balancer
5.	Performance	Enhance the performance by using IBM CDN	IBM Content Delivery Network

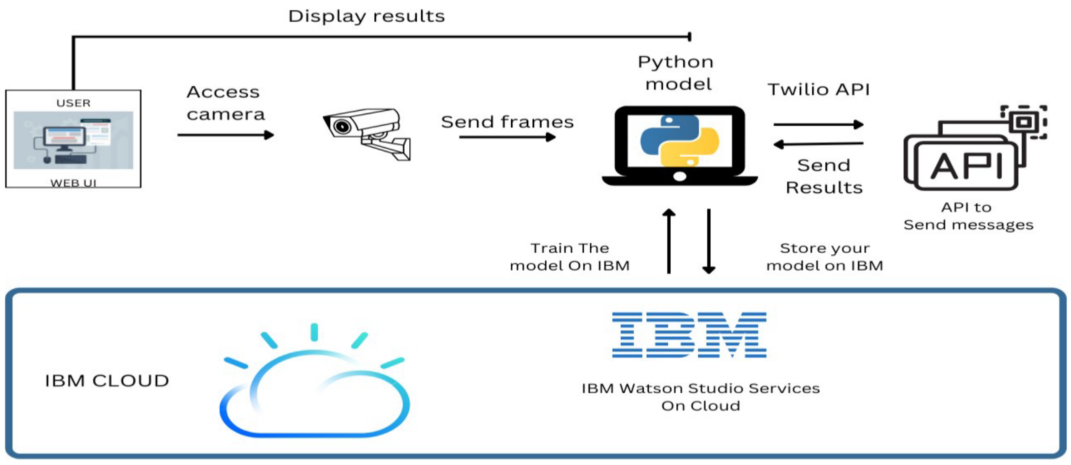
5. PROJECT DESIGN

5.1 Data Flow Diagrams

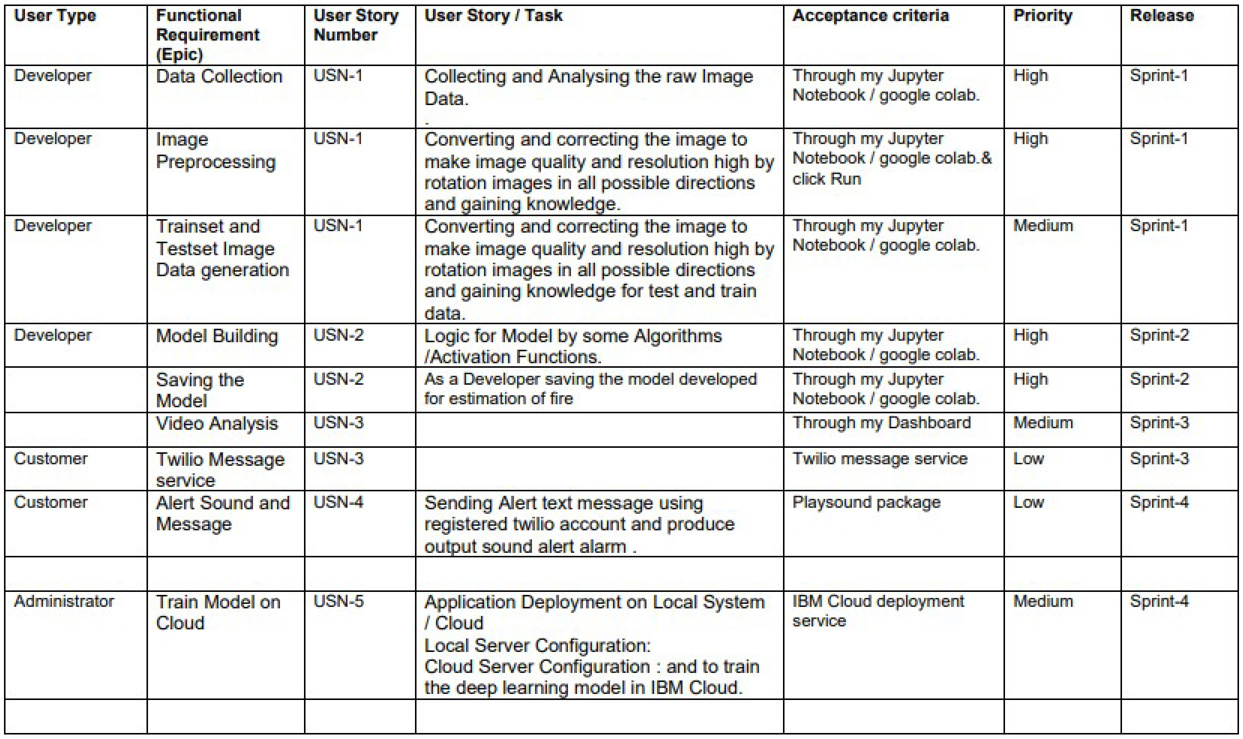


1. COLLECT DATA
2. EVALUATE DATA SET
3. IMPLEMENT ALGORITHMS
4. EVALUATE THE ACCURACY OF EACH ALGORITHMS
5. DISPLAY RESULTS

5.2 Solution & Technical Architecture



5.3 User Stories



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Developer	Data Collection	USN-1	Collecting and Analysing the raw Image Data. .	Through my Jupyter Notebook / google colab.	High	Sprint-1
Developer	Image Preprocessing	USN-1	Converting and correcting the image to make image quality and resolution high by rotation images in all possible directions and gaining knowledge.	Through my Jupyter Notebook / google colab.& click Run	High	Sprint-1
Developer	Trainset and Testset Image Data generation	USN-1	Converting and correcting the image to make image quality and resolution high by rotation images in all possible directions and gaining knowledge for test and train data.	Through my Jupyter Notebook / google colab.	Medium	Sprint-1
Developer	Model Building	USN-2	Logic for Model by some Algorithms /Activation Functions.	Through my Jupyter Notebook / google colab.	High	Sprint-2
	Saving the Model	USN-2	As a Developer saving the model developed for estimation of fire	Through my Jupyter Notebook / google colab.	High	Sprint-2
	Video Analysis	USN-3		Through my Dashboard	Medium	Sprint-3
Customer	Twilio Message service	USN-3		Twilio message service	Low	Sprint-3
Customer	Alert Sound and Message	USN-4	Sending Alert text message using registered twilio account and produce output sound alert alarm .	Playsound package	Low	Sprint-4
Administrator	Train Model on Cloud	USN-5	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration : and to train the deep learning model in IBM Cloud.	IBM Cloud deployment service	Medium	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S
		USN-2	As a user, I will receive confirmation email once I have registered for the application usage.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S
Sprint-2	Input	USN-3	Whenever the fire is detected, the information is given to the database.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S
Sprint-2		USN-4	When it is the wildfire then the alarming system is activated.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S
Sprint-3	Output	USN-5	And the alarm also sent to the corresponding departments and made them know that the wildfire is erupted.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S
Sprint-4	Action	USN-6	Required actions will be taken in order to control erupted wildfire by reaching as early as possible to the destination with thehelp of detecting systems.	20	High	CHINNADURAI.I VISHANTH.R ARUNKUMAR.M KADHARALI.S

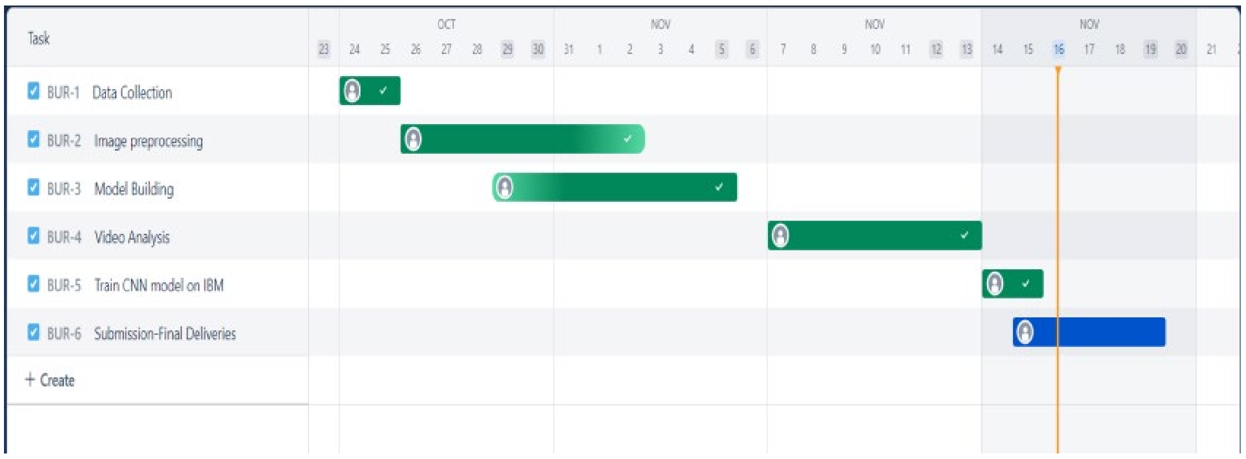
6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date(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

The following table shows the sprint works assigned to the members along with the priority and story points assigned with the functional requirements with regards to user story.

6.3 Reports from JIRA

Burndown Chart:



Task
BUR-1 Data Collection
BUR-2 Image preprocessing
BUR-3 Model Building
BUR-4 Video Analysis
BUR-5 Train CNN model on IBM
BUR-6 Submission-Final Deliveries
+ Create

7. CODING & SOLUTION

7.1 Feature 1

In Feature 1 module we have made data collection and Image preprocessing for and Model training.

importing Required Libraries:

```
import keras
from keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import numpy as np
batch_size = 32
```

image resizing and preprocessing :

```
train_datagen = ImageDataGenerator(
    shear_range=0.2,
    rotation_range=180,
    zoom_range=0.2,
    horizontal_flip=True,
)

val_datagen = ImageDataGenerator(
    rescale=1./255
)
train_generator = train_datagen.flow_from_directory(
    'train_set/',
    target_size=(150, 150),
    batch_size=batch_size,
    class_mode='binary'
)

val_generator = val_datagen.flow_from_directory(
    'test_set/',
    target_size=(150, 150),
    batch_size=batch_size,
    class_mode='binary'
)
```

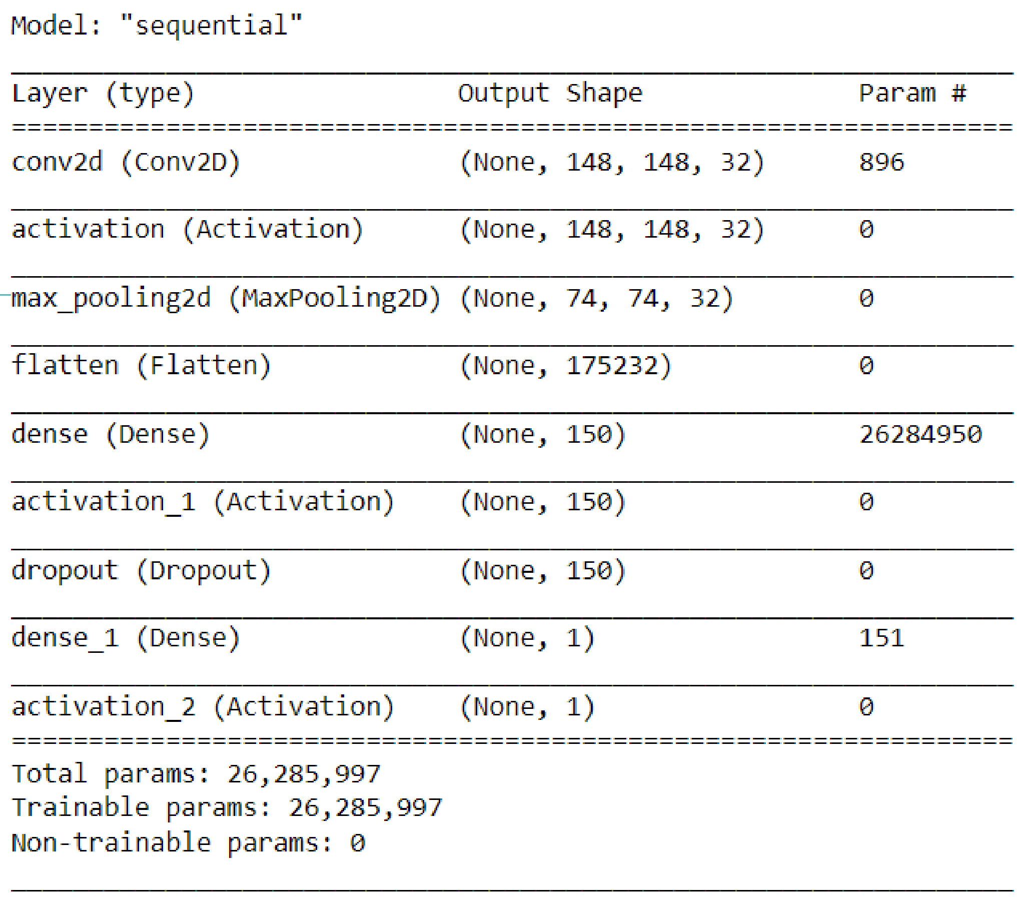
Creating the sequential model :

```
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Activation
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers import Dense

model=Sequential()
model.add(Convolution2D(32,(3,3),input_shape=(150,150,3))) #Convolutional 2D Layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2))) # MaxPooling Layer
model.add(Flatten())          #Flatten Layer to make a array
model.add(Dense(150))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation('sigmoid'))
model.compile(
    loss='binary_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

Model summary :

model.summary()

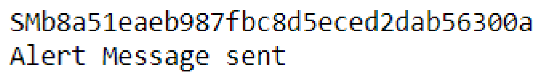


```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 148, 148, 32)      896
_________________________________________________________________
activation (Activation)      (None, 148, 148, 32)      0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 175232)            0
_________________________________________________________________
dense (Dense)                (None, 150)               26284950
_________________________________________________________________
activation_1 (Activation)    (None, 150)               0
_________________________________________________________________
dropout (Dropout)            (None, 150)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 151
_________________________________________________________________
activation_2 (Activation)    (None, 1)                 0
=================================================================
Total params: 26,285,997
Trainable params: 26,285,997
Non-trainable params: 0
_________________________________________________________________
```

7.2 Feature 2

```
from keras.models import load_model
from keras.preprocessing import image
import numpy as np
import cv2
from PIL import Image, ImageOps
model=load_model("forest1.h5")
from twilio.rest import Client
from playsound import playsound
model=load_model('forest1.h5')
video=cv2.VideoCapture(0)
name=['forest','with fire']
account_durai='ACca0e8bb11699d2957d67c979ca84b68a'
auth_token='bcb5f3850ef4b7ed263f60efc9acecdb' client
=Client(account_durai,auth_token)
message=client.messages \
.create(
body='-------Forest Fire is detected,Stay Alert !!!--------',
from_='+19457581434',to='+919043062227')
```

```
print(message.sid), print("Alert Message sent")
```



```
SMb8a51eaeb987fbc8d5eced2dab56300a
Alert Message sent
```

8. TESTING

8.1 Test Cases & User Acceptance Testing

Testing with input video recording from user end:

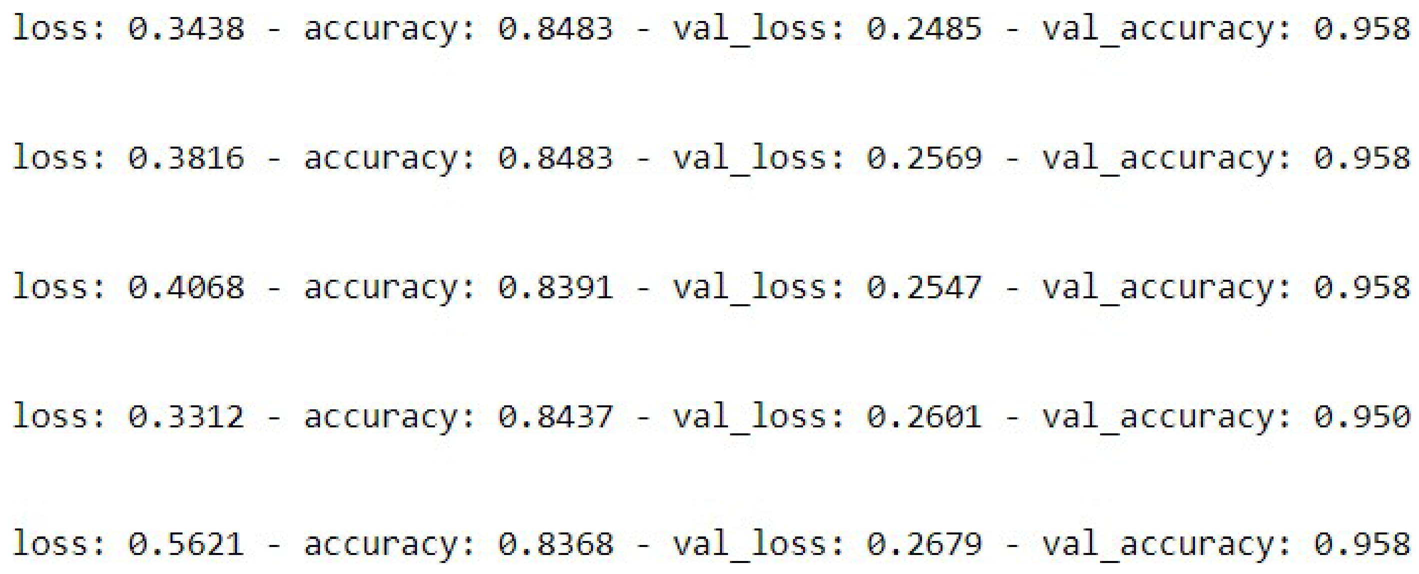
```
import cv2
import numpy as np
from keras.preprocessing import image
from keras.models import load_model
from twilio.rest import Client
from playsound import playsound
model=load_model('forest1.h5')
video=cv2.VideoCapture(0)
name=['forest','with fire']
while(True):
    ret,frame=video.read()
    cv2.imshow('frame',frame)
    cv2.imwrite('image.jpg',frame)
    img=image.load_img('image.jpg',target_size=(64,64))
    x=image.img_to_array(img)
    x=np.expand_dims(x,axis=0)
    pred=model.predict(x)
    index=np.argmax(pred)
    if index==0:
        account_durao='ACca0e8bb11699d2957d67c979ca84b68a'
        auth_token='bcb5f3850ef4b7ed263f60efc9acecdb'
        client
        =Client(account_durai,auth_token)
        message=client.messages \
        .create(body='-------Fire is detected,Stay Alert !!! ------- ',
                from_='+19457581434',to='+919043062227')
        print(message.sid)
        print('Fire detected')
        print("Alert Message sent!")
        playsound('tornado-siren.mp3')

    else:
        print('No Danger')
        cv2.imshow("image.jpg",frame)
        if cv2.waitKey(2) & 0xff == ord('a'):
            break
video.release()
cv2.destroyAllWindows()
```

```
account_durai = 'AC04fd8c4ea21f7599b004db5c72066eef'
auth_token = '48a23af63a81a9fe85bf6e600f3668f4'
client = Client(account_durai, auth_token)
message = client.messages \
.create(
body='Forest fire is detected , stay alert',
from_='+16075363954',
to='+919043062227'
)
print(message.durai)
```

9.RESULTS

9.1 Performance Metrics



```
loss: 0.3438 - accuracy: 0.8483 - val_loss: 0.2485 - val_accuracy: 0.958

loss: 0.3816 - accuracy: 0.8483 - val_loss: 0.2569 - val_accuracy: 0.958

loss: 0.4068 - accuracy: 0.8391 - val_loss: 0.2547 - val_accuracy: 0.958

loss: 0.3312 - accuracy: 0.8437 - val_loss: 0.2601 - val_accuracy: 0.950

loss: 0.5621 - accuracy: 0.8368 - val_loss: 0.2679 - val_accuracy: 0.958
```

10. ADVANTAGES & DISADVANTAGES

Advantages

- Easily detect and Estimate the Forest Fire.
- Most Accurate
- Flexible Model which can give maximized outcome
- No Specific Requirements needed to implement the model

Disadvanatges

- Training model is time consuming process.
- Error in Cv can cause damage to camera
- Access of camera are prohibited due to personal issues

11. CONCLUSION

Thus we have constructed a model that can can identify the effects of the forest fire and it can analyse the forest fire by advanced AI techniques and CNN Algorithm then the Prediction model is Checked and then the model is connected with Twilio account credentials of the Developer consisting of phone numbers of the persons in the surroundings of the people in the area of easy forest fire zone then an security sound alert system is developed to make a alert sound which is downloaded from internet then the entire model is deployed to the IBM Cloud account that we have created was made with the studies we have done.

12. FUTURE SCOPES

- It can be developed as a Web or Android Application.
- In future Alternate Advanced technologies can be Implemented.
- The Identification and tracking system can be implemented if possible.

13. APPENDIX

13.1 Source Code link

Source Code

```
from google.colab import drive
drive.mount('/content/drive')
Mounted at /content/drive
```

In [ ]:

```
!pip install tensorflow
!pip install opencv-python
!pip install opencv-contrib-python
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simpl
e/
Requirement already satisfied: tensorflow in /usr/local/lib/python3.7/dist-packages (2.9.2)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.7/dist-packages (from
tensorflow) (14.0.6)
Requirement already satisfied: tensorflow-estimator<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.
7/dist-packages (from tensorflow) (2.9.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in /usr/local/lib/python3.7/dist-packa
ges (from tensorflow) (1.1.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tenso
rflow) (1.15.0)
Requirement already satisfied: tensorboard<2.10,>=2.9 in /usr/local/lib/python3.7/dist-packages
(from tensorflow) (2.9.1)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from tenso
rflow) (3.1.0)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tenso
rflow) (1.21.6)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from ten
sorflow) (1.14.1)
Requirement already satisfied: flatbuffers<2,>=1.12 in /usr/local/lib/python3.7/dist-packages (f
rom tensorflow) (1.12)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in /usr/local/lib/python3.7/dist-packages (
from tensorflow) (3.19.6)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.7/
dist-packages (from tensorflow) (0.27.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from te
nsorflow) (1.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from tensorf
low) (21.3)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow) (3.3.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from
tensorflow) (2.1.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from tensor
flow) (57.4.0)
Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist-packages
 (from tensorflow) (2.9.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist-packages (fr
om tensorflow) (0.2.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist-packages (fr
om tensorflow) (1.50.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from
 tensorflow) (1.6.3)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.7/dist-packages (fr
om tensorflow) (0.4.0)
```

```
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist-package
s (from tensorflow) (4.1.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (fro
m astunparse>=1.6.0->tensorflow) (0.38.3)
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h
5py>=2.9.0->tensorflow) (1.5.2)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from t
ensorboard<2.10,>=2.9->tensorflow) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7
/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-pa
ckages (from tensorboard<2.10,>=2.9->tensorflow) (1.8.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist
-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.4.6)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist-packages (fr
om tensorboard<2.10,>=2.9->tensorflow) (2.23.0)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from t
ensorboard<2.10,>=2.9->tensorflow) (1.0.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist-packages (
from tensorboard<2.10,>=2.9->tensorflow) (2.14.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (
from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.2.8)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages
(from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (5.2.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from goo
gle-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist-package
s (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist-packages
 (from markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (4.13.0)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importl
ib-metadata>=4.4->markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (3.10.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (f
rom pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.4.8)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (fro
m requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2022.9.24)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3
.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from
 requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requ
ests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2.10)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from r
equests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (
3.2.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-package
s (from packaging->tensorflow) (3.0.9)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simpl
e/
Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66
)
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from ope
ncv-python) (1.21.6)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simpl
e/
Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.7/dist-packages (
4.6.0.66)
```

```
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from ope
ncv-contrib-python) (1.21.6)
```

In [ ]:

```
import tensorflow as tf
import numpy as np
from tensorflow import keras
import os
import cv2
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
```

In [ ]:

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

In [ ]:

```
train_dataset =
train.flow_from_directory("/content/drive/MyDrive/ibm/Dataset/Dataset/train_set",
                                          target_size=(128,128),
                                          batch_size = 32,
                                          class_mode = 'binary' )
Found 436 images belonging to 2 classes.
```

In [ ]:

```
test_dataset = test.flow_from_directory("/content/drive/MyDrive/ibm/Dataset/Dataset/test_set",
                                          target_size=(128,128),
                                          batch_size = 32,
                                          class_mode = 'binary' )
Found 320 images belonging to 2 classes.
```

In [ ]:

```
test_dataset.class_indices
```

Out[ ]:

```
{'forest': 0, 'with fire': 1}
```

In [ ]:

```
#to define linear initialisation import sequential
from keras.models import Sequential
#to add layer import Dense
from keras.layers import Dense
#to create convolution kernel import convolution2D
from keras.layers import Convolution2D
#import Maxpooling layer
from keras.layers import MaxPooling2D
#import flatten layer
from keras.layers import Flatten
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:

```
model = keras.Sequential()
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
```

In [ ]:

```
model.add(Dense(150,activation='relu'))

model.add(Dense(1,activation='sigmoid'))
```

In [ ]:

```
model.compile(loss = 'binary_crossentropy',
              optimizer = "adam",
              metrics = ["accuracy"])
```

In [ ]:

```
r = model.fit(train_dataset, epochs = 5, validation_data = test_dataset)
Epoch 1/5
14/14 [==============================] - 327s 24s/step - loss: 0.5417 - accuracy: 0.7294 - val_l
oss: 0.2729 - val_accuracy: 0.9219
Epoch 2/5
14/14 [==============================] - 31s 2s/step - loss: 0.2936 - accuracy: 0.8739 - val_los
s: 0.1726 - val_accuracy: 0.9187
Epoch 3/5
14/14 [==============================] - 30s 2s/step - loss: 0.1855 - accuracy: 0.9335 - val_los
s: 0.2235 - val_accuracy: 0.8938
Epoch 4/5
14/14 [==============================] - 30s 2s/step - loss: 0.2020 - accuracy: 0.9083 - val_los
s: 0.1444 - val_accuracy: 0.9406
Epoch 5/5
14/14 [==============================] - 29s 2s/step - loss: 0.1585 - accuracy: 0.9266 - val_los
s: 0.1118 - val_accuracy: 0.9688
```

In [ ]:

```
predictions = model.predict(test_dataset)
predictions = np.round(predictions)
10/10 [==============================] - 8s 839ms/step
```

In [ ]:

```
predictions
```

Out[ ]:

```
array([[1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [0.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [1.],
       [0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [1.],
       [0.],
       [0.],
       [1.],
       [0.],
       [0.],
       [1.],
       [1.]], dtype=float32)
```

In [ ]:

```
print(len(predictions))
320
```

In [ ]:

```
model.save("/content/drive/MyDrive/ibm/music/forest1.h5")
```

In [ ]:

```
#import load_model from keras.model
from keras.models import load_model
#import image class from keras
import tensorflow as tf
from tensorflow.keras.preprocessing import image
#import numpy
import numpy as np
#import cv2
import cv2
```

In [ ]:

```
model = load_model("/content/drive/MyDrive/ibm/music/forest1.h5")
```

In [ ]:

```
def predictImage(filename):
  img1 = image.load_img(filename,target_size=(128,128))
  Y = image.img_to_array(img1)
  X = np.expand_dims(Y,axis=0)
  val = model.predict(X)
  print(val)
  if val == 1:
    print(" fire")
  elif val == 0:
      print("no fire")
```

In [ ]:

```
predictImage("/content/drive/MyDrive/ibm/Dataset/Dataset/test_set/with
fire/Forest_fire_MNRF_esize_IMG_6743.jpg")
1/1 [==============================] - 0s 118ms/step
[[1.]]
 fire
```

In [ ]:

```
!pip install twilio
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simpl
e/
Collecting twilio
  Downloading twilio-7.15.2-py2.py3-none-any.whl (1.4 MB)
     |████████████████████████████████| 1.4 MB 4.2 MB/s
```

```
Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from t
wilio) (2.23.0)
Collecting PyJWT<3.0.0,>=2.0.0
  Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from twilio) (202
2.6)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requ
ests>=2.0.0->twilio) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (fro
m requests>=2.0.0->twilio) (2022.9.24)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from
 requests>=2.0.0->twilio) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3
.7/dist-packages (from requests>=2.0.0->twilio) (1.24.3)
Installing collected packages: PyJWT, twilio
Successfully installed PyJWT-2.6.0 twilio-7.15.2
```

In [ ]:

```
!pip install playsound
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simpl
e/
Requirement already satisfied: playsound in /usr/local/lib/python3.7/dist-packages (1.3.0)
```

In [ ]:

```
#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
```

In [ ]:

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

In [ ]:

```
account_durai = 'AC153cd036f84d1a7a72baf185e662b651'
auth_token = '********************************'
client = Client(account_durai, auth_token)

message = client.messages \
    .create(
         body='Forest fire  go to saved places',
         from_='+13605838876',
         to='+919043062227'
     )

print(message.sid)
SM35132eb94a5fb50e21844149b2b85005
```

```
import cv2
              import threading
              import playsound
              from twilio.rest import Client


              fire_cascade = cv2.CascadeClassifier('fire_detection.xml')

              vid = cv2.VideoCapture(0)
              runOnce = False
              Font = cv2.FONT_HERSHEY_SIMPLEX

              def play_alarm_sound_function():
                  playsound.playsound('fire_Alarm.mp3',True)
                  print("Fire alarm end")

              def send_Twilio_function():

                  SID = 'AC153cd036f84d1a7a72baf185e662b651'

                  Auth_Token = '***************************'

                  cl = Client(SID, Auth_Token)

                  cl.messages.create(body="Forest Fire Go To Saved Places", from_= '+13605838875',to='+919043062227')



              while(True):
                  Alarm_Status = False
                  ret, Forest = vid.read()
                  gray = cv2.cvtColor(Forest, cv2.COLOR_BGR2GRAY)
                  fire = fire_cascade.detectMultiScale(Forest, 1.2, 5)


                  for (x,y,w,h) in fire:
                      cv2.rectangle(Forest,(x-20,y-20),(x+w+20,y+h+20),(0,255,0),2)
                      roi_gray = gray[y:y+h, x:x+w]
                      roi_color = Forest[y:y+h, x:x+w]
                     cv2.putText(Forest,"Fire",(x,w),Font,2,(0,0,225),2,cv2.LINE_AA)

                      print("Fire alarm initiated")
                      threading.Thread(target=play_alarm_sound_function).start()  # To call alarm thread

                      if runOnce == False:
                          print("Twilio send initiated")
```

```
                threading.Thread(target=send_Twilio_function).start() # To call Twilio thread
                runOnce = True
            if runOnce == True:
                print("Twilio is already sent once")
                runOnce = True

        cv2.imshow('Forest', Forest)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            Break

    vid.release
    cv2.destroyAllWindows
import cv2
             import threading
             import playsound
             import smtplib

             fire_cascade = cv2.CascadeClassifier('train/fire_detection.xml')

             vid = cv2.VideoCapture(0)
             runOnce = False
             Font = cv2.FONT_HERSHEY_SIMPLEX

             def play_alarm_sound_function():
                 playsound.playsound('music/fire_Alarm.mp3',True)
                 print("Fire alarm end")

             def send_mail_function():

                 recipientmail = "19161681@saec.ac.in"
                 recipientmail = recipientmail.lower()

                 try:
                     server = smtplib.SMTP('smtp.gmail.com', 587)
                     server.ehlo()
                     server.starttls()
                     server.login("cdurai1321@gmail.com", 'zchgwcublcrobkqu')
                     server.sendmail('19161681@saec.ac.in', recipientmail, "WildFire forest ")
                     print("Alert mail sent sucesfully to {}".format(recipientmail))
                     server.close()

                 except Exception as e:
                     print(e)

             while(True):
                 Alarm_Status = False
                 ret, frame = vid.read()
                 gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
                 fire = fire_cascade.detectMultiScale(frame, 1.2, 5)


                 for (x,y,w,h) in fire:
                     cv2.rectangle(frame,(x-20,y-20),(x+w+20,y+h+20),(0,255,0),2)
                     roi_gray = gray[y:y+h, x:x+w]
                     roi_color = frame[y:y+h, x:x+w]
                    cv2.putText(frame,"Fire",(x,w),Font,2,(0,0,225),2,cv2.LINE_AA)

                     print("Fire alarm initiated")
                     threading.Thread(target=play_alarm_sound_function).start()  # To call alarm thread

                     if runOnce == False:
                         print("Mail send initiated")
                         threading.Thread(target=send_mail_function).start() # To call Email thread
                         runOnce = True
                     if runOnce == True:
                         print("Mail is already sent once")
                         runOnce = True

                 cv2.imshow('Forest', frame)
                 if cv2.waitKey(1) & 0xFF == ord('q'):
                     break

             vid.release
             cv2.destroyAllWindows
```