**PROJECT REPORT**

Project Name**: EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES.**

Team id: **PNT2022TMID07051**

Team members: Dharani. V

Anisha Singh. J

Arthi. A

Logavarshini. R

1. **INTRODUCTION**

# Project overview

Forest fires generally occur in wild areas due to human carelessness and change in an environment. They cause the environment and may result in human and wild animals deaths. So that, forest fires must be detected early to prevent greater damages. Wildﬁre, also called forest ﬁre, bush or vegetation ﬁre, can be described as any uncontrolled and non-prescribed combustion or burning of plants in a natural setting such as a forest, grassland, brush land or tundra, which consumes the natural fuels and spreads based on environmental conditions. Forest ﬁres are a major environmental issue, creating economic and ecological damage while endangering human lives. There are typically about more wildﬁres are occurring every year. It is diﬃcult to predict and detect Forest Fire in a 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 and also during the Fire due to its reliability and eﬃciency. This is a huge problem which needs to be tackled and thus through this project we provide a way to tackle the issue.

# Purpose

In this project we purpose is to detect the forest ﬁre earlier.

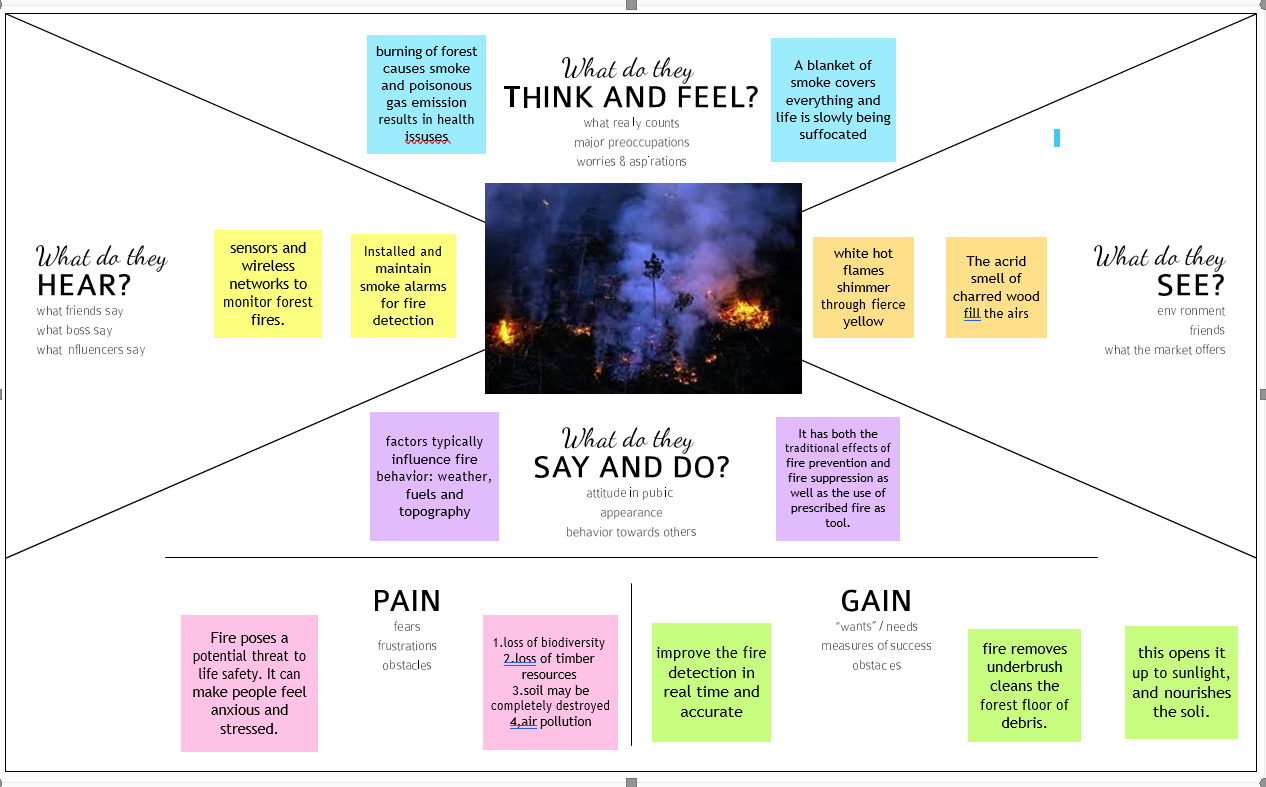
# LITERATURE SURVEY

* 1. **Reference**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **TITLE** | **AUTHOR** | **YEAR** |
| **1.** | A Review on Forest Fire Detection Techniques | Ahmad A. A. Alkhatib | 2013 |
| **2.** | Forest Fire Detection System using LoRa Technology. | Nicoleta Cristina Gaitani, Paula Hojbota | 2020 |
| **3.** | Wireless sensor network for forest fire detection. | Nelson Alberto Lizardo | 2020 |

# IDEATION & PROPOSED SOLUTION

* 1. **Empathy map**



* 1. **Proposed solution**

|  |  |  |
| --- | --- | --- |
| S.NO. | PARAMETER | DESCRIPTION |
| 1. | PROBLEM STATEMENT (PROBLEM TO BE SOLVED) | Detection of forest fire and smoke in wildland areas is done through remote sensing-based methods such as satellites, high-resolution static cameras fixed on the ground, and unmanned aerial  vehicles (UAVs). |
| 2. | IDEA/SOLUTION DESCRIPTION | Detection of forest fires using CO2 sensors  Existing detection methods such as satellite and optical systems can cover large areas; satellite systems identify infrared signatures, while optical systems look for  smoke plumes. |
| 3. | NOVELITY/UNIQUENESS | A fire detection system uses a smoke detector to detect a fire before it actually starts. An effective fire detection system eliminates damage by ensuring that a fire can be prevented before it even starts. A fire detector may also have a direct connection to an alarm monitoring centre |
| 4. | SOCIAL IMPACT/CUSTOMER SATISFACTION | Blocked roads and railway lines, electricity, mobile and land telephone lines cut, destruction of homes and industries, and the way of life of many communities are annual news stories and the balance of the catastrophe caused by fire results in a wealth of articles, editorials  and communications. |
| 5. | BUSINESS MODEL(REVENUE | Installing security and fire |

# Problem solution ﬁt

* The earlier results observed from the continuous monitoring of the temperature sensors selected included the decision of incorporating which sensor into the sensor node.
* Once the input image from the vedio 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.
* We classify images using a convolutional neural network and use other open CV tools.
* The outputs of all these sensors were monitored simultaneously to check the accuracy of these sensors for inclusion in the development of the node.
* From the results it was evident that SHT15 showed output much closer to the Hygrometer 1620 by Fluke both in terms of temperature and humidity, so SHT15 was considered to be the best choice for sensing temperature and humidity in the sensor node.
* In the second part of experiment analysis, the sensor node was tested in the simulation chamber and artificial environment of forest fire was created by setting up fire from the dried leaves, small dried tree branches and paper bits. Here the parameters monitored included Temperature, Humidity, Smoke, Methane, and Carbon Monoxide.

**NOVELTY:**

Makes use of real time monitoring and allows pre-cursors potential issues to be flagged up and immediately be addressed before major issues occur.

**BUSSINESS MODEL:**

Focuses more on sensor probes, wireless sensor network and machine learning which makes the deployment easier.

**FEASIBILITY OF SOLUTION:**

* Cost effective
* Accurate
* Reliable
* Economical

1. **REQUIREMENT ANALYSIS**

# Functional requirement

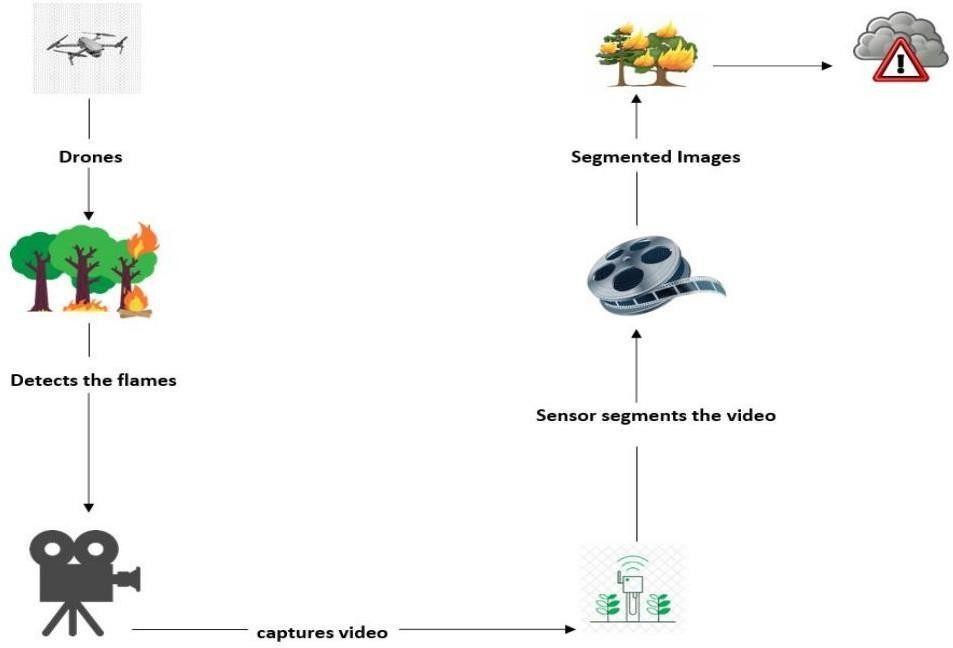
The system shall send a notification to the admin when it recognizes a fire in the image given the system shall take real inputs of satellite images and determine whether the image contains a fire or not The system shall be able to take images with a variety of sizes and convert it to one fixed image to be used throughout the application The system shall run as a service on either a Windows or Linux operating system. In the event that the computer on which the system is running shuts down, the system service should start automatically when the computer restarts Detection of forest fire and smoke in wildland areas is done through remote sensing-based methods such as satellites, high-resolution static cameras fixed on the ground, and unmanned aerial vehicles (UAVs).

* 1. **Non-Functional requirement**

Wireless Sensor Networks (WSNs) can be used for many applications, such as industrial automatic control, remote environmental monitoring, and target tracking. The similar system is promising applications in forest fires can make a real-time monitoring and detection. Wireless sensor network consists of numerous small nodes in most situations, which small nodes are deployed in remote and inaccessible hostile environments or over large geographical areas. The large number of small nodes sense environmental changes and report them to cluster head node over network architect, which the deployment and maintenance should be easy and scalable. A new approach for forest fire monitoring and detection was present in this paper, which it is by using data aggregation in wireless sensor network. The proposed approach can provide faster and efficiently reaction to forest fires while consuming economically WSN's energy, which has been validated and evaluated in extensive simulation experiments.

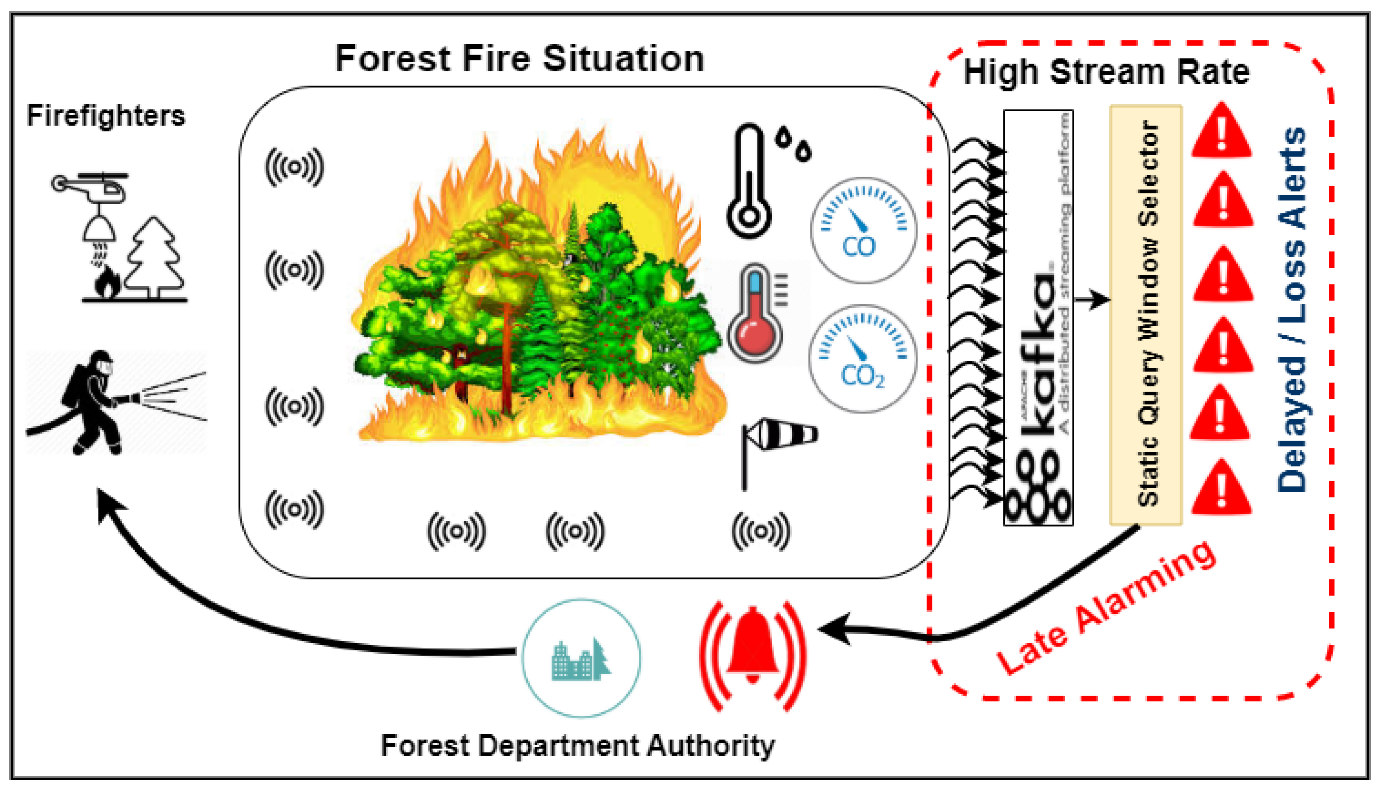
# PROJECT DESIGN

* 1. **Data Flow Diagram**



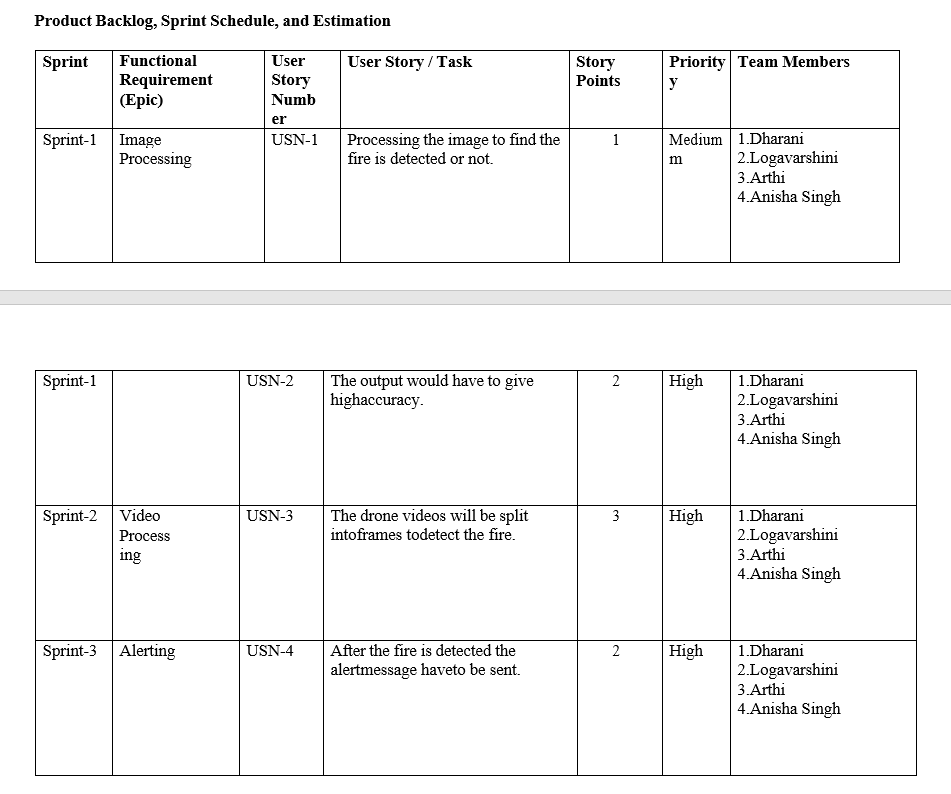
# Solution Architecture

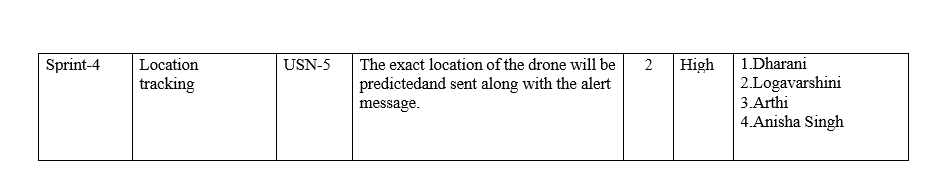
* The solution architecture helps ensure that a new system will fit the existing enterprise environment.
* To perform this task, a solution architect has to understand how all parts of the business model work together including processes, operating systems, and application architectures.
* A solution architecture is an architectural description of a specific solution. Solution Architecture combine guidance from different enterprise architecture viewpoints as well as from the enterprise solution architecture.



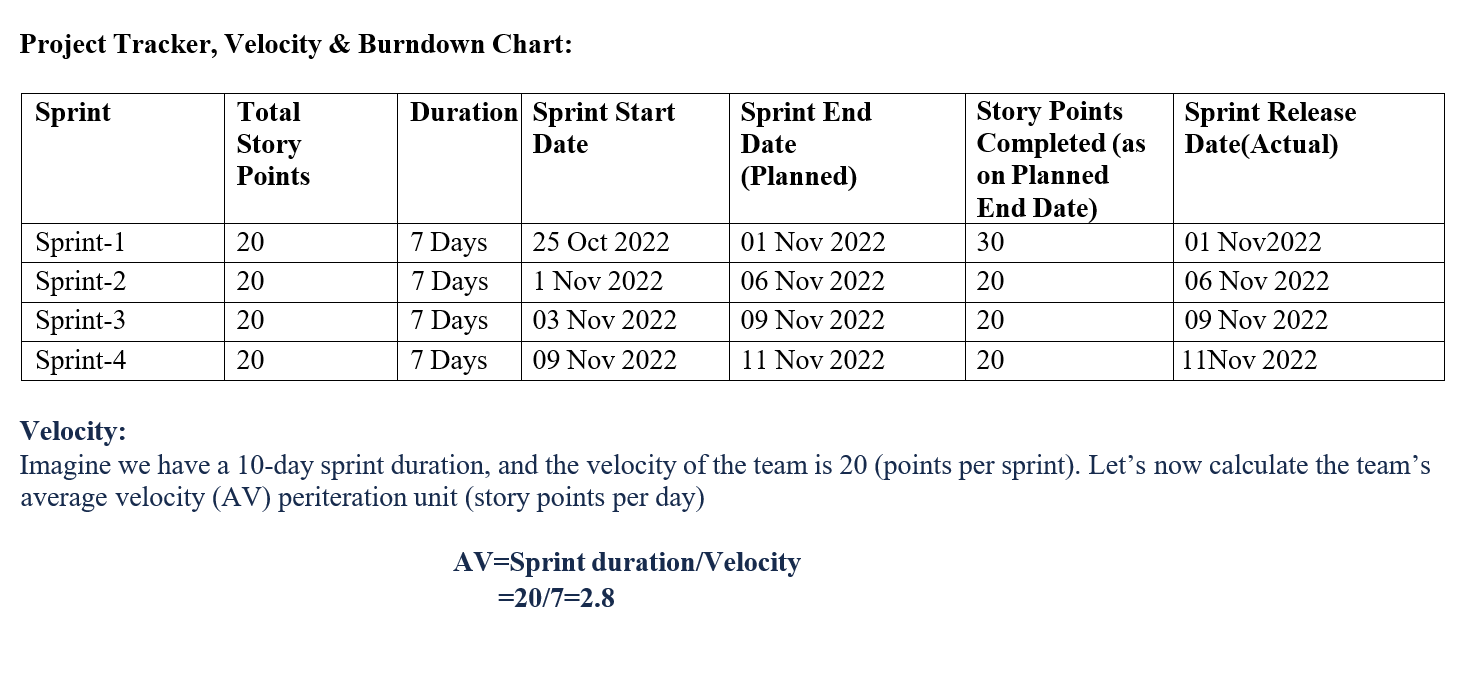
1. **PROJECT PLANNING & SCHEDULING**

# Sprint Planning & Estimation

****

****

* 1. **Sprint delivery schedule**

****

# COLLECTION OF DATATSET:



**CODE:**

#!/usr/bin/env python # **coding**: utf-8 # In[1]:

import zipfile # In[2]:

import tensorflow as tf # In[3]:

with zipfile.ZipFile('archive.zip', 'r') as zip\_ref: zip\_ref.extractall()

# In[4]: **#IMAGE PROCESSING** #

In[5]: import keras # In[6]:

from keras.preprocessing.image import ImageDataGenerator # In[7]:

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

) # In[8]: x\_train = train\_datagen.flow\_from\_directory(r'./Dataset/Dataset/train\_set',targe t

\_size=(128,128),

# In[9]: x\_test

=

batch\_size=32, class\_mode='binary')

test\_datagen.flow\_from\_directory(r'./Dataset/Dataset/test\_set',target\_ s ize=(128,128),

In[10]:

x\_test.class\_indices

# FEATURE 2 :

batch\_size=32, class\_mode='binary') #

### MODEL BUILDING, VIDEO ANALYSIS AND TRAINING THE CNN MODEL

In this phase, the model for detecting the forest fire is built and the video analysis for detecting fire is made. The Convolutional Neural Network model is trained.

### CODE :

# In[11]: **#MODEL BUILDING**

# In[12]: from keras.models import Sequential from keras.layers import



Dense from keras.layers import Convolution2D from keras.layers import MaxPooling2D from keras.layers import Flatten import warnings warnings.filterwarnings('ignore') # In[13]:

model = Sequential() # In[14]: model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='r e lu')) # In[15]:

model.add(MaxPooling2D(pool\_size=(2,2))) # In[16]:

model.add(Flatten()) # In[17]:

model.add(Dense(150,activation='relu')) # In[18]:

model.add(Dense(1,activation='sigmoid')) # In[19]: model.compile(loss='binary\_crossentropy', optimizer="adam", metrics=["accuracy"]) # In[20]:

model.fit(x\_train, epochs=5, validation\_data=x\_test, batch\_size=32) # In[21]:

model.save("forest1.h5") # In[22]:

import numpy as np # In[23]:

predictions = model.predict(x\_test) predictions = np.round(predictions) # In[24]: predictions # In[25]:

print(len(predictions)) # In[26]: 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[27]:

model = load\_model(r'forest1.h5') # In[28]: 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[30]: predictImage(r'C:\Users\PADMAVATHY\Desktop\IBM- project\Dataset\Dataset\test\_set\with fire\louisiana\_forest\_fire.jpg') # In[31]:

**#OPENCV For Video Processing**

# In[32]:

pip install twilio # In[33]:

pip install playsound # In[34]:

#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[35]:

#load the saved model model = load\_model(r'forest1.h5') #define video

video = cv2.VideoCapture('VideoCapture.mp4') #define the features name = ['forest','with forest'] #

In[36]:

# Creating An Account in Twilio Service # In[37]:

account\_sid='ACb64c545bdd1f4134f579761c63d979d0' auth\_token='b1d8a216257673a32d5a60b6e6ce2a9e' client=Client(account\_sid,auth\_token) message=client.messages .create( body='Forest Fire is detected, stay alert', from\_='+18583305228', to='+919361637171'

) print(message.sid) # In[37]:

**#Sending Alert Message**

# In[38]:

from keras.preprocessing import image # In[38]:

from matplotlib import pyplot as plt #import load model from keras.model from keras.models import load\_model

#import image from keras

from tensorflow.keras.preprocessing import image img1 = image.load\_img(r'C:\Users\PADMAVATHY\Desktop\IBM- project\Dataset\Dataset\test\_set\with fire\deerfire\_high\_res\_edit.jpg',target\_size=(128,128)) Y = image.img\_to\_array(img1) x = np.expand\_dims(Y,axis=0) val = model.predict(x) plt.imshow(img1) plt.show() if val==1: print('Forest fire') # In[40]:

if val==1:

print('Forest fire') from twilio.rest import Client print('Forest fire')

account\_sid='ACb64c545bdd1f4134f579761c63d979d0' auth\_token='b1d8a216257673a32d5a60b6e6ce2a9e' client=Client(account\_sid,auth\_token) message=client.messages

.create( body='Forest fire is detected, stay alert!',from\_='+18583305228', to='+919361637171')

print(message.sid) print("Fire detected") print("SMS Sent!")

elif val==0:

print('No Fire')

#!/usr/bin/env python # coding: utf-8

# In[4]: import os, types import pandas as pd

from botocore.client import Config import ibm\_boto3

def iter (self): return 0 # @hidden\_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos\_client = ibm\_boto3.client(service\_name='s3', ibm\_api\_key\_id='TZF0HDYpFwXDzuDIpXfWmCnIpwByLziPqcp9CfptkWaL', ibm\_auth\_endpoint="https://iam.cloud.ibm.com/oidc/token", config=Config(signature\_version='oauth'), endpoint\_url='https://s3.private.us.cloud-object-

storage.appdomain.cloud')

bucket = 'cnn-donotdelete-pr-chivj41em9ccer' object\_key

= 'archive.zip'

streaming\_body\_1 = cos\_client.get\_object(Bucket=bucket, Key=object\_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.

# Please read the documentation of ibm\_boto3 and pandas to learn more about the possibilities to load the data.

# ibm\_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/ # pandas documentation: <http://pandas.pydata.org/># In[5]: from io import BytesIO import zipfile unzip=zipfile.ZipFile(BytesIO(streaming\_body\_1.read()),'r') file\_paths=unzip.namelist() for path in file\_paths:

unzip.extract(path) # In[6]:

pwd # In[10]: import os filenames=os.listdir('/home/wsuser/work/Dataset/Dataset/train\_set'

) # In[11]:

get\_ipython().system('pip install libgl1-mesa-dev') import tensorflow as tf import numpy as np from tensorflow import keras import os from tensorflow.keras.preprocessing.image import ImageDataGenerator from tensorflow.keras.preprocessing import image # In[12]:

#Define the parameters/arguments for ImageDataGenerator class # In[13]:

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[14]:

# Applying ImageDataGenerator functionality to trainset # In[15]: x\_train =

train.flow\_from\_directory("/home/wsuser/work/Dataset/Dataset/train\_set "

,

In[16]:

target\_size=(64,64), batch\_size = 32, class\_mode = 'binary' ) #

#Applying ImageDataGenerator functionality to testset # In[17]: x\_test =

test.flow\_from\_directory("/home/wsuser/work/Dataset/Dataset/test\_set",

target\_size=(64,64), batch\_size = 32, class\_mode = 'binary' )

# In[18]:

x\_test.class\_indices # In[19]:

from keras.models import Sequential from keras.layers import Dense from keras.layers import Convolution2D from keras.layers import MaxPooling2D from keras.layers import Flatten import warnings warnings.filterwarnings('ignore')

# In[20]:

**#Initializing the model**

# In[21]:

model =Sequential() # In[22]: **# Add CNN Layer** # In[23]: model.add(Convolution2D(32,(3,3),input\_shape=(64,64,3),activation='rel u

'))

model.add(MaxPooling2D(pool\_size=(2,2))) model.add(Flatten()) # In[24]: **# Add**

**Hidden Layer** # In[25]:

model.add(Dense(150,activation='relu')) model.add(Dense(1,activation='sigmoid')) # In[26]:

# Configure the learning process # In[27]:

model.compile(loss = 'binary\_crossentropy',

optimizer = "adam", metrics

= ["accuracy"])

# In[28]:

model.save("/home/wsuser/work/archive(1)/forest1.h5") # In[29]:

predictions = model.predict(x\_test) predictions = np.round(predictions) # In[30]: predictions # In[31]: print(len(predictions)) #

In[33]:

from keras.models import load\_model import tensorflow as tf from tensorflow.keras.preprocessing import image import numpy as np # In[34]:

model = load\_model("/home/wsuser/work/archive(1)/forest1.h5") # In[35]: def predictImage(filename): img1 = image.load\_img(filename,target\_size=(64,64))

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[37]:

predictImage("/home/wsuser/work/Dataset/Dataset/test\_set/wit h fire/19464620\_401.jpg") # In[38]:

pip install twilio # In[39]:

pip install playsound # In[40]:

import numpy as np from keras.preprocessing import image from keras.models import load\_model from twilio.rest import Client from playsound import playsound # In[41]: model = load\_model(r'/home/wsuser/work/archive(1)/forest1.h5') name = ['forest','with forest']

# In[42]:

#Creating An Account In Twilio Service # In[43]:

pip install pygobject # In[44]:

def message(val):

if val==1: from twilio.rest import Client print('Forest fire')

account\_sid='AC6be2d13a80de59f51a5fe3ba2bf9d6f1' auth\_token='00ac87e22f4bbc807a00a5ca30eedd1e' client=Client(account\_sid,auth\_token) message=client.messages .create( body='forest fire is detected, stay alert',

#use twilio free number from\_='+14793912961', #to number to='+917358579433')

print(message.sid) print("Fire detected") print("SMS Sent!")

elif val==0: print('No Fire')

# In[46]: from matplotlib import pyplot as plt from keras.models import load\_model from tensorflow.keras.preprocessing import image img1 = image.load\_img('/home/wsuser/work/Dataset/Dataset/test\_set/with fire/deerfire\_high\_res\_edit.jpg',target\_size=(64,64))

Y = image.img\_to\_array(img1) x = np.expand\_dims(Y,axis=0) val = model.predict(x) plt.imshow(img1) plt.show() message(val) # In[48]: img2 =

image.load\_img('/home/wsuser/work/Dataset/Dataset/test\_set/forest/beau t iful\_morning\_mountain\_forest\_scenery\_free\_stock\_photos\_picjumbo\_DSC021 7

6.jpg',target\_size=(64,64)) Y = image.img\_to\_array(img2) x = np.expand\_dims(Y,axis=0) val = model.predict(x) plt.imshow(img2) plt.show() message(val) # In[49]:

from ibm\_watson\_machine\_learning import APIClient wml\_credentials={"url":"https://us- south.ml.cloud.ibm.com","apikey":"TFXoHzN3M76f8UM68mdo\_MshGtF2Dk1H56fJ 6

7oDagbV"} client=APIClient(wml\_credentials) # In[50]:

def guid\_from\_space\_name(client,space\_name): space=client.spaces.get\_details() return(next(item for item in space['resources']if

item['entity']["name"]==space\_name)['metadata']['id']) # In[51]:

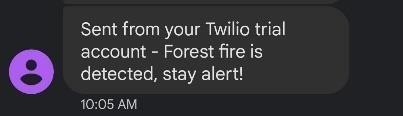
space\_uid=guid\_from\_space\_name(client,'imageclassification') print("Space UID= "+space\_uid)

# In[52]:

client.set.default\_space(space\_uid) # In[53]:

client.software\_specifications.list() # In[54]: software\_spec\_uid=client.software\_specifications.get\_uid\_by\_name("tens o rflow\_1.15-py3.6") software\_spec\_uid # In[55]: keras

# OUTPUT SCREENSHOT:



# TESTING AND RESULTS

* + - * 1. **Performance Testing**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Parameter** | **Values** | **Screenshot** |
| 1. | Model Summary | This project is created using AI and ML, which detects the forest fire and sends the alert message to the respective individual using Twilio API.  Keras and Open CV are also effectively used in the project. | *Figure 6Alert Message* |
| 2. | Accuracy | Training Accuracy – 95% Validation Accuracy -62% |  |
| 3. | Confidence Score (Only Yolo Projects) | Confidence Score – 97% |  |

Defeat Analysis:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severit y1** | **Severit y2** | **Severit y3** | **Severit y4** | **Subtotal** |
| By Design | 5 | 1 | 1 | 1 | 8 |
| Duplicate | 1 | 0 | 3 | 0 | 4 |
| External | 2 | 3 | 0 | 1 | 6 |
| Fixed | 7 | 2 | 4 | 10 | 23 |
| Not Reproduced | 0 | 0 | 0 | 0 | 0 |
| Skipped | 0 | 0 | 1 | 1 | 2 |
| Won'tFix | 0 | 3 | 2 | 1 | 6 |
| Totals | 15 | 9 | 11 | 14 | 4  9 |

## 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 | 5 | 0 | 0 | 5 |
| Client Application | 30 | 0 | 0 | 30 |
| Security | 2 | 0 | 0 | 2 |
| Out source Shipping | 3 | 0 | 0 | 3 |
| Exception Reporting | 9 | 0 | 0 | 9 |
| Final Report Output | 4 | 0 | 0 | 4 |
| Version Control | 2 | 0 | 0 | 2 |

1. **ADVANTAGES & DISADVANTAGES**

Wireless sensor network technology normally deploys large number of small, low cost sensors fairly densely that can observe and influence the physical world around them by gathering physical information, transforming it into electrical signals, sending it to a remote location to do some analysis, and deploying the results in different applications. By this way there is no need to build towers or set up complicated communication links such as microwave and satellite. It can be deployed anywhere even in inaccessible places.

This technology can provide real-time monitoring, where it can provide information at the ignition instance or at very small delays, depending on the node used in wake-up/sleep schedule.

This technology works on short communication links fashion. As a result, more accurate information with less delay can be provided for the fire fighters.

Using this technology for forest fire application requires a large number of randomly deployed nodes to provide a reliable network. Based on what has been discussed above, it can define the key issues of this network for this application in the following points.

1. Localization: all the previous work used a GPS or fixed the nodes in a known place.
2. Coverage: the nodes deployed randomly a full coverage almost impossible.
3. Network life span: For sensor nodes working on batteries, it is impossible to go back to each node in the forest and recharge it again.
4. Fire detection method: this is the heart of the application; it should be precise and reliable. In the previous work they used other kinds of sensors like IP cameras or other technologies like satellite images or superoptical tower cameras. They integrated the sensor nodes information with data bases, weather stations, fuel and weather indexing models, and many other models and processing techniques to reduce the number of false alarms. In order to detect the fire, each node was provided with many sensors to get the environment parameters such as temperature, wind speed, relative humidity, and fire flickering to define a fire incident in the first place.

# CONCULSION

# 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 takes place within both the sensor node and at the base station. For the system, to fit any weather condition, climatic condition, or area, a threshold ratio is introduced for analysis within the sensor node. In the case of node deployment, it can be mounted at any place in the forest even if there is no preinstalled network connectivity, as the transceiver module is based on dedicated built-in network infrastructure. Because of the primary power supply provided by rechargeable batteries with a secondary solar power supply, a solution is readily implementable as a standalone system for prolonged periods. The proposed system incorporated with the communication infrastructure alerted the relevant authorities with lower latency than the existing systems during the numerous test trials conducted in real tropical forest sites.

# FUTURE SCOPE:

# 

# Evolution emerges in the processing, computation, and algorithms. This strives many researchers to pay attention in many domains where they work in the processing of surveillance video streams so that abnormal or unusual actions could be detected. The usage of UAVs is recommended in the detection of forest fire due to the high mobility and ensures the coverage areas at various altitudes and locations at a low cost. Hence, an efficient and scalable UAV is used for detection. This work aims in developing the 3D model for the captured scene. As our future works, focus to meet practical detection and meet the necessity of early detection including the generation of the mixed reality model of the forest fire area that gives more information, and prevention analysis will be made easy. The 3D modelling techniques presented in this paper can also be extended to various natural disaster prediction models.

# GitHub:

https://github.com/IBM-EPBL/IBM-Project-28698-1660115295

# 