# Detecting Fire Combustions in Forest

**Artificial Intelligence Externship program by SmartBridge**

**Team Name: Tech Tigers**

## Team Members

1. M.Nagarjuna – 17MIS7162
2. A.Sai Haneesh Reddy – 17MIS7160
3. S.Giridhar – 17MIS7131
4. T.Tharun Raju – 17MIS7094

## Overview:

The problem with forest fires is that the forests are usually remote, abandoned/unmanaged areas filled with trees, dry and parching wood, leaves, and so forth that act as a fuel source. These elements form a highly combustible material and represent the perfect context for initial-fire ignition and act as fuel for later stages of the fire. The fire ignition may be caused through human actions like smoking or barbeque parties or by natural reasons such as high temperature in a hot summer day or a broken glass working as a collective lens focusing the sun light on a small spot for a length of time thus leading to fire-ignition.

## Software Requirements:

- Jupyter Notebook
- IBM Cloud
- IBM Watson Studio
- Web Cam
- Python Libraries

## Hardware Requirements:

- Laptop of RAM 8GB
- I3/I5 Processor
- Stable Internet

## Methodology:

Optical sensors or camera systems in general need to be improved in order to reduce the number of false alarms due to various dynamic phenomena such as wind-tossed trees, cloud shadows, reflections, and human activity. The difficulties of processing landscape images are due to their varying nature and to the large number of dynamic events that may appear under various illuminating conditions depending on weather, distance, time of day, masking objects, and so forth. These events produce dynamic envelopes, which are not always caused by motion, and consist of time-varying gray levels of connected pixels in several image regions.

## Dataset:

- Forest-fire dataset from Kaggle

## Usecases:

- Used to save many Living Animals in the Forest.
- Used to detect fires in Forest

## Code:

```
#import keras library

import keras

#import ImageDataGenerator class from keras

from keras.preprocessing.image import ImageDataGenerator

#Define the parameters /arguments 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)

#: Applying ImageDataGenerator functionality to trainset.

#give the path of training images folder

x_train =
train_datagen.flow_from_directory(r'C:/Users/Lenovo/Downloads/forest-
fire/Dataset/Dataset/train_set/',

                                     target_size = (128,128),

                                     batch_size = 32,

                                     class_mode = 'binary')

#: Applying ImageDataGenerator functionality to testset.

#give the path of testing images folder
```

```
x_test =
test_datagen.flow_from_directory(r'C:/Users/Lenovo/Downloads/forest-
fire/Dataset/Dataset/test_set/',
                                  target_size = (128,128),
                                  batch_size = 32,
                                  class_mode = 'binary')
'''import model building libraries'''


#To define linear intialisation import Sequential
from keras.models import Sequential
#To add layers 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')
#intializing the model
model =Sequential()
#configure the learning process
model.compile(loss = 'binary_crossentropy',
            optimizer = "adam",

            metrics = ["accuracy"])

#Training the model

model.fit_generator(x_train,steps_per_epoch=14,

                epochs=10,validation_data=x_test,

                validation_steps=4)

#import load_model from keras.model

from keras.models import load_model

#import image class from keras

from keras.preprocessing import image

#import numpy

import numpy as np

#import cv2

import cv2

#load the saved model

model = load_model("forest1.h5")

#give any random image path

img = image.load_img(r'C:\Users\Lenovo\Downloads\forest-
fire\Dataset\Dataset\test_set\with fire\19464620_401.jpg',target_size =
(128,128))

x = image.img_to_array(img)

#expand the image shape

x = np.expand_dims(x,axis = 0)

import cv2
```

```
#import facevec
import numpy as np
import smtplib
from keras.preprocessing import image
from keras.models  import load_model
from twilio.rest import Client


model = load_model(r'C:\Users\Lenovo\Downloads\forest-fire\forest1.h5')
video = cv2.VideoCapture(0)
name = ['forest','with fire']


while(1):
   success, frame = video.read()
   cv2.imwrite(r"C:\Users\Lenovo\Downloads\Detecting-Forest-Combustion-in-Forests-main\Detecting-Forest-Combustion-in-Forests-main\Training\image.jpg",frame)
   img = image.load_img(r"C:\Users\Lenovo\Downloads\Detecting-Forest-Combustion-in-Forests-main\Detecting-Forest-Combustion-in-Forests-main\Training\image.jpg",target_size = (64,64))
   x  = image.img_to_array(img)
   x = np.expand_dims(x,axis = 0)
   pred = model.predict_classes(x)
   p = pred[0]
   print(pred)
```

```
    cv2.putText(frame, "predicted  class = "+str(name[p]), (100,100),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,0), 1)



    pred = model.predict_classes(x)
    if pred[0]==1:
        account_sid = 'ACa56253bf3f2e2918bxxxxxxx'
        auth_token = 'a10cb957a1b8bc17abbxxxxxxx'
        client = Client(account_sid, auth_token)

        message = client.messages \
        .create(
         body='Forest Fire is detected, stay alert',
         from_=' +1503512xxxx', #twilio free number
         to='+919160xxxxxx')
        print(message.sid)

        print('Fire Detected')
        print ('SMS sent!')
        break
    else:
        print("no danger")
       #break
    cv2.imshow("image",frame)
```

```
 if cv2.waitKey(1) & 0xFF == ord('a'):
     break
video.release()
cv2.destroyAllWindows()
```