**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