

Image Preprocessing

#importing keras library

```
import keras
```

#importing the image data generator

```
from matplotlib import pyplot as plt
```

```
from keras.preprocessing.image import ImageDataGenerator
```

#Defining the parameter for image generator 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 image data generator functionality to train set

```
x_train=train_datagen.flow_from_directory(r'D:\IBM\Dataset\train_set',  
target_size=(128, 128), batch_size=32, class_mode='binary')
```

Found 436 images belonging to 2 classes.

#Applying image data generator functionality to test set

```
x_test=test_datagen.flow_from_directory(r'D:\IBM\Dataset\test_set',  
target_size=(128, 128), batch_size=32, class_mode='binary')
```

Found 121 images belonging to 2 classes.

Model Building

#To define linear intialisation import Sequential

```
from keras.models import Sequential
```

#To add layers import Dense

```
from keras.layers import Dense
```

#To creat Convolution kernal 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')
```

#initializing the model

```
model=Sequential()
```

#add convolution layer

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

#add maxpooling layer

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

#add convolution layer

```
model.add(Convolution2D(64, (3,3), activation='relu'))
```

```

#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add convolution layer
model.add(Convolution2D(128,(3,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add convolution layer
model.add(Convolution2D(128,(3,3),activation='relu'))
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
#add flatten layer
model.add(Flatten())

model.add(Dense(512,activation='relu'))
model.add(Dense(1,activation='sigmoid'))

#configuring the learning process
model.compile(loss='binary_crossentropy',optimizer="adam",metrics=["accuracy"])

#Training the model
r=model.fit(x_train,epochs=10,validation_data=x_test)

Epoch 1/10
14/14 [=====] - 84s 6s/step - loss: 0.6308 - accuracy: 0.7018 - val_loss: 0.2621 - val_accuracy: 0.8760
Epoch 2/10
14/14 [=====] - 47s 3s/step - loss: 0.3211 - accuracy: 0.8716 - val_loss: 0.0939 - val_accuracy: 0.9917
Epoch 3/10
14/14 [=====] - 47s 3s/step - loss: 0.1914 - accuracy: 0.9243 - val_loss: 0.0372 - val_accuracy: 0.9917
Epoch 4/10
14/14 [=====] - 48s 3s/step - loss: 0.1565 - accuracy: 0.9450 - val_loss: 0.0691 - val_accuracy: 0.9669
Epoch 5/10
14/14 [=====] - 50s 4s/step - loss: 0.1557 - accuracy: 0.9427 - val_loss: 0.0328 - val_accuracy: 0.9917
Epoch 6/10
14/14 [=====] - 47s 3s/step - loss: 0.1514 - accuracy: 0.9358 - val_loss: 0.0433 - val_accuracy: 0.9917
Epoch 7/10
14/14 [=====] - 47s 3s/step - loss: 0.1670 - accuracy: 0.9335 - val_loss: 0.0590 - val_accuracy: 0.9752
Epoch 8/10
14/14 [=====] - 47s 3s/step - loss: 0.1394 - accuracy: 0.9358 - val_loss: 0.0332 - val_accuracy: 0.9917
Epoch 9/10
14/14 [=====] - 49s 4s/step - loss: 0.1704 - accuracy: 0.9289 - val_loss: 0.0519 - val_accuracy: 1.0000
Epoch 10/10

```

```
14/14 [=====] - 49s 3s/step - loss: 0.2227 -  
accuracy: 0.9106 - val_loss: 0.0602 - val_accuracy: 1.0000
```

```
#save the model  
model.save("forestalert.h5")
```

Video Analysis

```
from twilio.rest import Client  
  
#import load model from keras.model  
from keras.models import load_model  
#import image from keras  
from tensorflow.keras.preprocessing import image  
import numpy as np  
#import cv2  
import cv2  
#load the saved model  
model=load_model(r"forestalert.h5")  
img=image.load_img(r'D:\IBM\Dataset\test_set\with fire\  
Uttarakhand_forest_fire.jpeg')  
x=image.img_to_array(img)  
# res=cv2.resize(x,dsize=(150,150),interpolation=cv2.INTER_CUBIC)  
#expand the image shape  
x=np.expand_dims(x,axis=0)  
  
from logging import WARNING  
#import opencv library  
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  
#import playsound package  
from playsound import playsound  
  
import cv2  
import numpy as np  
from matplotlib import pyplot as plt  
import tensorflow  
from tensorflow.keras.preprocessing import image  
from keras.models import load_model  
  
# Create a VideoCapture object and read from input file  
# If the input is the camera, pass 0 instead of the video file name  
video = cv2.VideoCapture(r'C:\Users\Dell\Downloads\Forest trees.mp4')  
name=['no fire','with fire']
```

```

while(1):
    success,frame = video.read()
    cv2.imwrite("image.jpg",frame)
    img = tensorflow.keras.utils.load_img("image.jpg",target_size =
(128,128))
    x = image.img_to_array(img)
    x = np.expand_dims(x,axis = 0)
    pred = model.predict(x)
    pred = pred[0][0]
    if pred > 0.5:
        pred = 1
    else :
        pred = 0
    print(pred)
    cv2.putText(frame,"predicted class = "+str(name[pred]),
(50,50),cv2.FONT_HERSHEY_SIMPLEX,1,(255,255,255),1)
    if pred==1:
        account_sid='ACab5b7ac22466b88a9cda7cf5414b750a'
        auth_token='a1a37c4aceac1ccf7b4a5a0b36bc62e9'
        client=Client(account_sid,auth_token)
        message=client.messages \
        .create(
            body='forest fire is detected,stay alert',
            #use twilio free number
            from_='+17088477470',
            #to number
            to='+918825826199')
        print(message.sid)
        print("Fire detected")
        print("SMS Sent!")
        from playsound import playsound
        playsound(r"C:\Users\Dell\Downloads\alarm.mp3")
        cv2.imshow('image',frame)
        if cv2.waitKey(0)&0xFF == ord('a'):
            break
    else:
        print("No Danger")
        cv2.imshow('image',frame)
        if cv2.waitKey(0)&0xFF == ord('a'):
            break
    video.release()
    cv2.destroyAllWindows()

```

1/1 [=====] - 0s 37ms/step
0
No Danger

WITH FIRE

Here a video with forest fire is given as input

```
# Create a VideoCapture object and read from input file
# If the input is the camera, pass 0 instead of the video file name
video = cv2.VideoCapture(r'C:\Users\Dell\Downloads\fire in
forest.mp4')
name=['forest','with fire']

while(1):
    success,frame = video.read()
    cv2.imwrite("image.jpg",frame)
    img = tensorflow.keras.utils.load_img("image.jpg",target_size =
(128,128))
    x = image.img_to_array(img)
    x = np.expand_dims(x,axis = 0)
    pred = model.predict(x)
    pred = pred[0][0]
    if pred > 0.5:
        pred = 1
    else :
        pred = 0
    print(pred)
    cv2.putText(frame,"predicted class = "+str(name[pred]),
(50,50),cv2.FONT_HERSHEY_SIMPLEX,1,(255,255,255),1)
    if pred==1:
        account_sid='ACab5b7ac22466b88a9cda7cf5414b750a'
        auth_token='a1a37c4aceac1ccf7b4a5a0b36bc62e9'
        client=Client(account_sid,auth_token)
        message=client.messages \
            .create(
                body='forest fire is detected,stay alert',
                #use twilio free number
                from_='+17088477470',
                #to number
                to='+918825826199')
        print(message.sid)
        print("Fire detected")
        print("SMS Sent!")
    from playsound import playsound
```

```
playsound(r"C:\Users\Dell\Downloads\alarm.mp3")
cv2.imshow('image', frame)
if cv2.waitKey(0)&0xFF == ord('a'):
    break
else:
    print("No Danger")
    cv2.imshow('image', frame)
    if cv2.waitKey(0)&0xFF == ord('a'):
        break
video.release()
cv2.destroyAllWindows()

1/1 [=====] - 0s 84ms/step
1
SM4322be4e3e8f86b3711c28cc826212ce
Fire detected
SMS Sent!
```