

EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

MODEL BUILDING

PREDICTIONS

Date	09 November 2022
Team ID	PNT2022TMID13306
Project Name	Emerging Methods for Early Detection of Forest Fires

Importing The ImageDataGenerator Library

```
import 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

```
x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/  
Dataset/train_set',target_size=(128,128),batch_size=32,  
class_mode='binary')
```

Found 436 images belonging to 2 classes.

Applying ImageDataGenerator functionality to testset

```
x_test=test_datagen.flow_from_directory(r'/content/drive/MyDrive  
/ Dataset/test_set',target_size=(128,128),batch_size=32,  
class_mode='binary')
```

Found 121 images belonging to 2 classes.

Import model building libraries

```
#To define Linear initialisation 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')
```

Initializing the model

```
model=Sequential()
```

Add CNN 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 flatten layer
```

```
model.add(Flatten())
```

Add Hidden Layer

```
#add hidden layer
```

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

```
#add output layer
```

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

Configure the learning process

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

Train the model

```
model.fit_generator(x_train,steps_per_epoch=14,epochs=10,validation_data=x_test,validation_steps=4)
```

Epoch 1/10

14/14 [=====] - 97s 7s/step - loss: 1.3060 - accuracy: 0.7775 - val_loss: 0.5513 -

val_accuracy: 0.8512
 Epoch 2/10
 14/14 [=====] - 26s 2s/step - loss: 0.3178 - accuracy: 0.8807 - val_loss: 0.1299 - val_accuracy: 0.9421
 Epoch 3/10
 14/14 [=====] - 26s 2s/step - loss: 0.2226 - accuracy: 0.9106 - val_loss: 0.1311 - val_accuracy: 0.9421
 Epoch 4/10
 14/14 [=====] - 31s 2s/step - loss: 0.1836 - accuracy: 0.9174 - val_loss: 0.1129 - val_accuracy: 0.9339
 Epoch 5/10
 14/14 [=====] - 30s 2s/step - loss: 0.1675 - accuracy: 0.9243 - val_loss: 0.0925 - val_accuracy: 0.9669
 Epoch 6/10
 14/14 [=====] - 26s 2s/step - loss: 0.1884 - accuracy: 0.9289 - val_loss: 0.1287 - val_accuracy: 0.9339
 Epoch 7/10
 14/14 [=====] - 28s 2s/step - loss: 0.1724 - accuracy: 0.9335 - val_loss: 0.0926 - val_accuracy: 0.9752
 Epoch 8/10
 14/14 [=====] - 26s 2s/step - loss: 0.1510 - accuracy: 0.9404 - val_loss: 0.0757 - val_accuracy: 0.9752
 Epoch 9/10
 14/14 [=====] - 26s 2s/step - loss: 0.173 - 2
 accuracy: 0.9174 - val_loss: 0.0537 - val_accuracy: 0.9835

Epoch 10/10 14/14 [=====] - 26s
0.154 -
2s/step - loss: 6
accuracy: 0.9312 - val_loss: 0.0573 -
val_accuracy: 0.9835
<keras.callbacks.History at 0x7f05d66a9c90>

Save The Model

```
model.save("forest1.h5")
```

Predictions

```
#import load_model from keras.model
```

```
from keras.models import load_model
```

```
#import image class from keras
```

```
from tensorflow.keras.preprocessing import image
```

```
#import numpy
```

```
import numpy as np
```

```
#import cv2
```

```
import cv2
```

```
#load the saved model
```

```
model = load_model("forest1.h5")
```

```
img=image.load_img(r'/content/drive/MyDrive/Dataset/test_set/forest /  
0.48007200_1530881924_final_forest.jpg') x=image.img_to_array(img)
```

```
res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER_CUBIC)
#expand the image shape x=np.expand_dims(res,axis=
0) pred= model.predict(x)

1/1 [=====] - 0s 126ms/step pred
array([[0.]], dtype=float32)
```