EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

MODEL BUILDING

**TRAINING THE MODEL**

|  |  |
| --- | --- |
| **Date** | 04 November 2022 |
| **Team ID** | PNT2022TMID50173 |
| **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,rot ati on\_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/MyDriv e/ 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=[ "ac curacy"])

# Train the model

model.fit\_generator(x\_train,steps\_per\_epoch=14,epochs=10,validation\_ da ta=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:

accuracy: 0.9174 - val\_loss: 0.0537 - val\_accuracy: 0.9835

Epoch 10/10

14/14 [==============================] - 26s

2s/step - loss:

accuracy: 0.9312 - val\_loss: 0.0573 - val\_accuracy: 0.9835

<keras.callbacks.History at 0x7f05d66a9c90>

0.173 - 2

0.154 - 6