INTRODUCTION

1.1 Overview:-

Fires represent a constant threat to ecological systems, infrastructure and human lives. Past has witnessed multiple instances of fires. With the faster and faster urbanization process, more and more high-rise buildings appear around us. This also can make the frequency of fire increase and bring great losses to people's lives and property. In areas where fire would pose an unreasonable threat to property, human life or important biological communities, efforts should be made to reduce dangers of fire. As the damage caused by fires is so tremendous that the early fire detection is becoming more and more important.

1.2 Problem Statement:-

Fire is common issue happening and the damage caused by these type of incidents is tremendous toward nature and human interest. Due to this the need for application for fire detection has increases in recent years. Fire detection algorithm based on image processing techniques which is compatible in surveillance devices like CCTV, wireless camera. The algorithm uses RGB color model to detect the color of the fire which is mainly comprehended by the intensity of the component R which is red color. The growth of fire is detected using sober edge detection. Finally cnn algorithm was applied based on the results from the first technique and second technique to identify the region of interest of the fire.

LITERATURE REVIEW

2. LITERATURE SURVEY

Paper 1:

FIRE DETECTION ALGORITHM USING IMAGE PROCESSING TECHNIQUE
Publication: year 2015
Authors: - Kumarguru Poobalan, Siau-Chuin Liew

In this paper we understand the concept of fire detection. The Authors proposed a fire detection algorithm based on image processing techniques. The algorithm uses RGB color model to detect the color of the fire which is mainly comprehended by the intensity of the component R which is red color. The growth of fire is detected using sobel edge detection. Finally a color based segmentation technique was applied based on the results from the first technique and second technique to identify the region of interest (ROI) of the fire.

Paper 2:

FIRE DETECTION USING IMAGE PROCESSING
Publication: year 2010
Authors: - Turgay Celik

In this paper, a new image-based real-time fire detection method was proposed which is based on computer vision techniques. The proposed method consists of three main stages: fire pixel detection using color, moving pixel detection, and analyzing fire-colored moving pixels in consecutive frames to raise an alarm. The proposed fire color model achieves a detection rate of 99.88% on the ten tested video sequences with diverse imaging conditions.

Paper 3:

FIRE DETECTION ON A SURVEILLANCE SYSTEM USING IMAGE PROCESSING
Publication: year 2017
Authors: - Prof. Amit Hatekar, Saurabh Manwani, Gaurav Patil, Akshat Parekh

This project proposed a fire detection algorithm which is free from sensors as the ordinary fire detection systems contain. The objective of this project was to create a system which would be able to detect fire as early as possible from a live video feed. System is expected to detect fire while it is still small and has not grown to mammoth proportions. Also, the hardware is minimal and has been already existent in places, thus saving capital. It also saves cost by getting rid of expensive temperature and heat sensors etc. Based on the results produced, the system has proven to be effective at detecting fire. This system is an amalgamation of various fire detection algorithms.

Paper 4:

FIRE DETECTION SYSTEM USING IMAGE PROCESSING
Publication: year 2016

Authors: - Ansari Mohd, Salim Badruddin

This project, Fire Detection System has been developed using Image Processing and Matlab software. This system has the ability to apply image processing techniques to detect fire. This system can be used to monitor fire and has achieved 90% accuracy for single webcam

Paper 5:

IOT BASED FOREST FIRE DETECTION SYSTEM
Publication: year 2018
Authors: - M. Trinath Basu, Ragipati Karthik

It is substantially less demanding to stifle a fire when the beginning area is known, and keeping in mind that it is in its beginning periods. Data about the advance of flame is likewise profoundly profitable for dealing with the fire amid every one of its stages

Paper 6:

AUTOMATIC FIRE DETECTION
Publication: year 2010
Authors: - Majid Bahrepour, Nirvana Meratnia, Paul Havinga
In this paper previous work in fire detection domain were surveyed from different perspectives. Our interest for this literature survey is to identify which sensor combinations and algorithms can detect fires accurately and quickly.

Paper 7:

FIRE DETECTION ALGORITHM USING IMAGE PROCESSING TECHNIQUE
Publication: year 2018
Authors:-ANSARI MOHD SALIM BADRUDDIN

The aim of the project is to early detection of fire apart from preventive measures to reduce the losses due to hazardous fire. The project mainly is based on image processing and Arduino serial communication. In this project, at the user end, the fire images will be feeded in the form of video frames. These images will be further processed by using the software, MATLAB. The proposed system uses RGB and YCbCr color space. The advantage of using YCbCr color space is that it can separate the luminance from the chrominance more effectively than RGB color space. Along with this smoke, motion, area detection is also performed using its color characteristics**.**

Paper 8:

FIRE DETECTION ALGORITHM USING IMAGE PROCESSING TECHNIQUE
Publication: year 2015
Authors: - Muhammad Shazali Dauda, Usman Saleh Toro
The Arduino based fire alarm detection and control system was proposed. It is capable of automatically detecting heat in a given environment, sound an alarm, switch off mains of the building and also spray water to reduce the intensity of fire. The system uses a DHT 11 sensor, a buzzer, 5v DC (Direct Current) motor, a GSM (Global System for Mobile) Module sim800l to send SMS (Short Message Service) and a LCD screen 16X2 and Atmeg328p Microcontroller. At the end, the objectives of this project were achieved and the system worked effectively

Paper 9:

FIRE DETECTION ALGORITHM USING IMAGE PROCESSING TECHNIQUE
Publication: year 2015
Authors: - Trung Luong

The author has gained the valuable experience in the field of fire detection and alarm system from studying and conducting the project. The objectives of the project were to provide information on fire alarm system in Vietnam and Finland, to show the similarities and differences with systems in both countries. Secondly, in the practical part, the objective was to build a demo system to demonstrate how a basic fire alarm system works. To achieve the purpose of this thesis, the author studied the main standards on fire detection and alarm systems in Vietnam and Finland. For the practical part, Arduino Uno was used as the control unit with other necessary components. Upon completing this project, the author has demonstrated how a fire detection and alarm system works and analysed the system standards in the above-mentioned countries. Moreover, the fire alarm system using the Arduino Uno was tested and found to work successfully.

Paper 10:

FIRE DETECTION ALGORITHM USING IMAGE PROCESSING TECHNIQUE
Publication: year 2015
Authors: - . Ola Willstrand, Peter Karlsson, Jonas Brandt

This report summarises the work that has been conducted in a large project about fire detection and fire alarm systems in heavy vehicles. The main goal of the project has been to develop an international test standard for fire detection systems installed in engine compartments of heavy vehicles. For the purpose of defining a test method background information has been compiled regarding fire detection technologies, relevant standards and guidelines, research in the field, durability factors associated with the environment, typical fire scenarios and fire causes.

PROPOSED WORK/ METHODOLOGY

3 PROPOSED WORK / METHODOLOGY

3.1 Proposed Algorithm: -

Our proposed method, the first step in our method is to detect the colour of the fire which is mostly red in colour. Then we used the sobel edge detection on the original image to detect the edge of the fire while removing threshold which is less than 100. Then we applied the segmentation technique which used the combine the result from the first technique and second technique to separate the ROI of the fire from the background.

RGB Colour Model:-

A fire image can be described by using its color properties. There are three different element of color pixel: R, G and B. The color pixel can be extracted into these three individual elements R, G and B, which is used for color detection. RGB color model is used to detect red color information in image.

3.1.2: **Sobel Edge Detection:-**

The next step will be to use the sobel edge detector to detect the growth of fire within the images. This can be done by applying 3x3 mask to the images.

3.1.3: **Segmentation Technique:-**

The final technique used in this algorithm is segmentation technique which was used to segmented fire from the non-fire background. The first step done by this technique is to specific the colour range for segmented process in the ROI.

3.1.4**: Result and Analysis**:-

Finally validation was carried out to evaluate the algorithm based on the 1000 images. This validation process uses a truth model, with which the results was compared. The contingency table for the sensitivity and specificity analyses of the validation. The true positive (TP) and true negatives (TN) are correct classification. A false positive (FP) is when the outcome of the algorithm is incorrectly predicted, when the in reality it is actually present in the image. The accuracy of the algorithm specify the ability of the algorithm in detecting the ROI. Accuracy = TP/ (TP+TN)*100%

The efficiency test is given as Efficiency = (TN+TP/TN+TP+FN+FP)*100%

DESIGN AND IMPLEMENTATION

4 DESIGN AND IMPLEMENTATION

4.1 Environmental Set-up:-

This paper proposes fire detection model for detecting fire from datasets. The class imbalance problem is handled by finding legal as well as fraud transaction patterns for each customer by using frequent itemset mining. A matching algorithm is also proposed to find to which pattern (legal or fraud) the incoming transaction of a particular customer is closer and a decision is made accordingly. In order to handle the anonymous nature of the data, no preference is given to any of the attributes and each attribute is considered equally for finding the patterns. The performance evaluation of the proposed model is done on UCSD Data Mining Contest 2009 Dataset (anonymous and imbalanced) and it is found that the proposed model has very high fraud detection rate, balanced classification rate, Matthews correlation coefficient, and very less false alarm rate than other state-of-the-art classifiers.

4.1.1 HARDWARE REQUIREMENTS

- PROCESSOR : Intel® Core™ i5
- RAM : 8.00 GB
- HARD DISK : 64 GB or more

4.1.2 SOFTWARE REQUIREMENTS

- OPERATING SYSTEM: Windows 10
- TOOLS: Anaconda, Jupiter Notebook
- CODING LANGUAGE: Python

DATASET USED

The datasets used in this study are open source and freely available online. Here is the dataset is the form of Images. Dataset is a large collection of densly-labeled Images that show whether the image has fire or not.

WORK FLOW

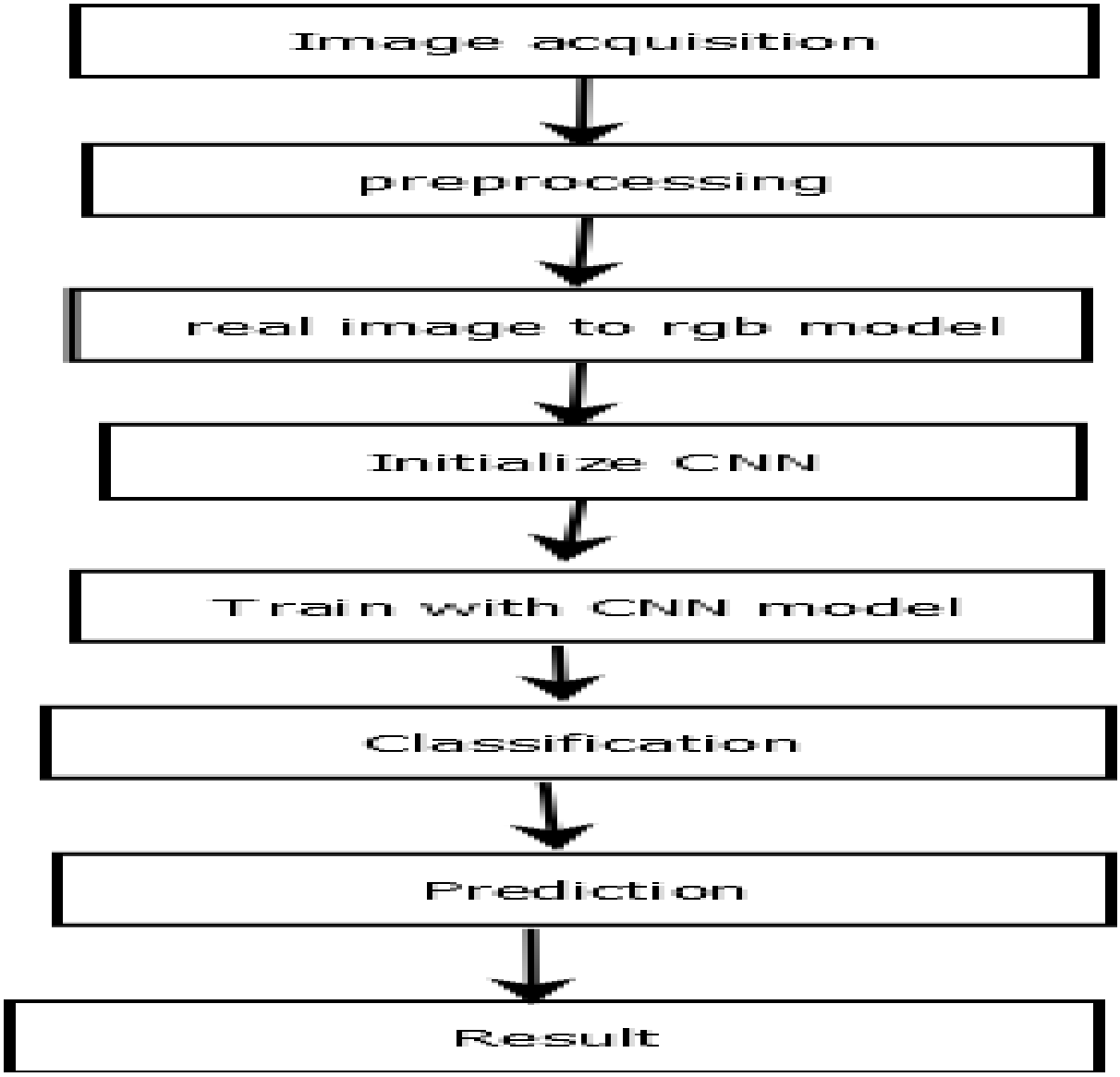
Image acquisition
preprocessing
real image to rgb model
Initialize CNN
Train with CNN model
Classification
Prediction
Result

RESULTS AND DISCUSSION

5 RESULTS AND DISCUSSION

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [56]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [57]:
```
tf.__version__
```

Out[57]: '2.8.0'

In [58]:
```
# Checking if i have GPU Enabled,then process willbe faster
tf.test.is_gpu_available()
```

Out[58]: False

In [ ]:
```
#creating more data from existing data
```

In [59]:
```
# 2 very important preprocessing is Resizing and Rescaling

batch_size=16

training_datagenarator= ImageDataGenerator(rescale=1./255,horizontal_flip=True,
    vertical_flip=True,shear_range=0.2,
    zoom_range=0.2,width_shift_range=0.2,
    height_shift_range=0.2,validation_split=0.1)
```

In [60]:
```
# dividing the data into training and validation
```

In [61]:
```
train=training_datagenarator.flow_from_directory('D:\Research\Fire-Detection',
                                                 target_size=(224, 224),color_mode='rgb',
                                        class_mode='binary', batch_size=batch_size,subset='training')

validation=training_datagenarator.flow_from_directory('D:\Research\Fire-Detection',
                                                 target_size=(224, 224),color_mode='rgb',
                                        class_mode='binary', batch_size=batch_size,subset='validation')
```

```
Found 586 images belonging to 4 classes.
Found 65 images belonging to 4 classes.
```

Run Code

In [62]:
```
# Initializing CNN
cnn=tf.keras.models.Sequential()

# adding first layer
cnn.add(tf.keras.layers.Conv2D(filters=32,kernel_size=3,padding='same',activation='relu',input_shape=[224,224,3]))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# adding second layer
cnn.add(tf.keras.layers.Conv2D(filters=64,kernel_size=3,padding='same',activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# adding third layer
cnn.add(tf.keras.layers.Conv2D(filters=256,kernel_size=3,padding='same',activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# Flattening
cnn.add(tf.keras.layers.Flatten())

# Fully connected layer
cnn.add(tf.keras.layers.Dense(units=128,activation='relu'))

# Output layers
cnn.add(tf.keras.layers.Dense(units=1,activation='sigmoid'))
```

In [63]:
```
cnn.summary()
```

In [64]:
```
checkpoint=tf.keras.callbacks.ModelCheckpoint('D:\Research\Fire-Detection',
                                              monitor='val_loss',mode="min",
                                              save_best_only=True)
callbacks=checkpoint
```

In [65]:
```
cnn.compile(optimizer='Adam',loss='binary_crossentropy',metrics=['accuracy'])

cnn.fit_generator(train,validation_data=validation,epochs=1,
                  steps_per_epoch=train.samples//batch_size,
                  validation_steps=validation.samples//batch_size,
                  callbacks=callbacks
                 )
```

```
<ipython-input-65-7e5d21895a19>:3: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Pl
ease use `Model.fit`, which supports generators.
  cnn.fit_generator(train,validation_data=validation,epochs=1,

36/36 [==============================] - ETA: 0s - loss: -624237.5000 - accuracy: 0.1632INFO:tensorflow:Assets written to: D:\R
esearch\Fire-Detection\assets
36/36 [==============================] - 108s 3s/step - loss: -624237.5000 - accuracy: 0.1632 - val_loss: -4245072.5000 - val_a
ccuracy: 0.1719
```

Out[65]:
```
<keras.callbacks.History at 0x1aa943003a0>
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_9 (Conv2D)           (None, 224, 224, 32)      896

 max_pooling2d_9 (MaxPooling  (None, 112, 112, 32)     0
 2D)

 conv2d_10 (Conv2D)          (None, 112, 112, 64)      18496

 max_pooling2d_10 (MaxPoolin  (None, 56, 56, 64)       0
 g2D)

 conv2d_11 (Conv2D)          (None, 56, 56, 256)       147712

 max_pooling2d_11 (MaxPoolin  (None, 28, 28, 256)      0
 g2D)

 flatten_3 (Flatten)         (None, 200704)            0

 dense_6 (Dense)             (None, 128)               25690240

 dense_7 (Dense)             (None, 1)                 129

=================================================================
Total params: 25,857,473
Trainable params: 25,857,473
Non-trainable params: 0
_________________________________________________________________
```

In [72]:
```
from tensorflow.keras.models import load_model
cnn=load_model('D:\Research\Fire-Detection')
```

In [67]:
```
cnn.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_9 (Conv2D)           (None, 224, 224, 32)      896

 max_pooling2d_9 (MaxPooling  (None, 112, 112, 32)     0
 2D)

 conv2d_10 (Conv2D)          (None, 112, 112, 64)      18496

 max_pooling2d_10 (MaxPoolin  (None, 56, 56, 64)       0
 g2D)

 conv2d_11 (Conv2D)          (None, 56, 56, 256)       147712

 max_pooling2d_11 (MaxPoolin  (None, 28, 28, 256)      0
 g2D)

 flatten_3 (Flatten)         (None, 200704)            0

 dense_6 (Dense)             (None, 128)               25690240

 dense_7 (Dense)             (None, 1)                 129
```

In [68]:
```
from tensorflow.keras.preprocessing import image
import numpy as np
import matplotlib.pyplot as plt
import PIL
```

In [92]:
```
image_for_testing=r'D:/Research/Fire-Detection/none-fire/3.jpg'
```

In [97]:
```
image_for_testing=r'D:/Research/Fire-Detection/none-fire/3.jpg'
```

In [98]:
```
test_image=image.load_img(image_for_testing,target_size=(224,224))
test_image=image.img_to_array(test_image)
test_image=test_image/255
test_image=np.expand_dims(test_image,axis=0)
result=cnn.predict(test_image)
if result[0][0]==0:
    print("there is fire")
else:
    print("there is no fire")
PIL.Image.open(image_for_testing)
```

```
there is no fire
```

98]:


```
plt.show()
```

5.1 CONCLUSIONS AND FUTURE SCOPE

We proposed a fire detection algorithm based on image processing techniques. The algorithm uses RGB colour model to detect the colour of the fire which is mainly comprehended by the intensity of the component R which is red colour. The growth of fire is detected using sobel edge detection. Finally a colour based segmentation technique was applied based on the results from the first technique and second technique to identify the region of interest (ROI) of the fire. The algorithm works very well when there is a fire outbreak. The overall accuracy of the algorithm is greater than 90%, indicating the effectiveness and usefulness of the algorithm. In future work, a real-time based algorithm could be consider as it might increase the efficiency of the algorithm which is currently 80.64%.

REFERENCES

6 REFERENCES

- https://www.kaggle.com/code/atulyakumar98/fire-detection
- https://www.researchgate.net/publication/285580944_FIRE_DETECTION_ALGORITHM_USING_IMAGE_PROCESSING_TECHNIQUES
- https://core.ac.uk/download/pdf/55305301.pdf
- https://www.sciencedirect.com/science/article/pii/S2214157X2030085X
- https://medium.com/@nauman.hanif/fire-detection-algorithm-using-image-processing-5cea78220dd5
- file:///C:/Users/gudiy/Downloads/sustainability-12-08899%20(1).pdf
- https://www.sciencedirect.com/topics/engineering/fire-detection
- https://www.kaggle.com/atulyakumar98/test-dataset
- https://www.youtube.com/watch?v=hxjreioqBeA&t=13s
- https://www.youtube.com/watch?v=mrUQlsZlO50&t=2175s

APPENDIX – SOURCE CODE

7. SOURCE CODE

```
import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

tf.__version__

tf.test.is_gpu_available()

batch_size=16

training_datagenarator= ImageDataGenerator(rescale=1./255,horizontal_flip=True,

    vertical_flip=True,shear_range=0.2,

    zoom_range=0.2,width_shift_range=0.2,

    height_shift_range=0.2,validation_split=0.1)

train=training_datagenarator.flow_from_directory('D:\Research Project\Fire-Detection',

 target_size=(224, 224),color_mode='rgb',

 class_mode='binary', batch_size=batch_size,subset='training')


validation=training_datagenarator.flow_from_directory('D:\Research  Project\Fire-Detection',
target_size=(224, 224),color_mode='rgb',

class_mode='binary', batch_size=batch_size,subset='validation')

# Initializing CNN

cnn=tf.keras.models.Sequential()

# adding first layer

cnn.add(tf.keras.layers.Conv2D(filters=32,kernel_size=3,padding='same',activation='relu',inp
ut_shape=[224,224,3]))

cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# adding second layer
```

```
cnn.add(tf.keras.layers.Conv2D(filters=64,kernel_size=3,padding='same',activation='relu'))

cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# adding third layer

cnn.add(tf.keras.layers.Conv2D(filters=256,kernel_size=3,padding='same',activation='relu'))

cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

# Flattening

cnn.add(tf.keras.layers.Flatten())

# Fully connected layer

cnn.add(tf.keras.layers.Dense(units=128,activation='relu'))

# Output layers

cnn.add(tf.keras.layers.Dense(units=1,activation='sigmoid'))

cnn.summary()

checkpoint=tf.keras.callbacks.ModelCheckpoint(r'D:\Research
Project\models\fire_nonfire_models.h5', monitor='val_loss',mode="min",

save_best_only=True)  callbacks=checkpoint

cnn.compile(optimizer='Adam',loss='binary_crossentropy',metrics=['accuracy'])

cnn.fit_generator(train,validation_data=validation,epochs=1,

 steps_per_epoch=train.samples//batch_size,

 validation_steps=validation.samples//batch_size,

 callbacks=callbacks

 )

from tensorflow.keras.models import load_model

cnn=load_model('D:\Research\Fire-Detection')
cnn.summary()
```

```
from tensorflow.keras.preprocessing import image

import numpy as np

import matplotlib.pyplot as plt

import PIL
image_for_testing=r'D:\Research Project\Fire-Detection\none-fire\9.jpg'

test_image=image.load_img(image_for_testing,target_size=(224,224))

test_image=image.img_to_array(test_image)

test_image=test_image/255

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

result=cnn.predict(test_image)

#result=cnn.predict_class(test_image)

Catagories=['fire','no fire']

image_show=PIL.Image.open(image_for_testing)

plt.imshow(image_show)

plt.title(Catagories[int(result[0][0]==0)])

plt.show()
image_for_testing=r'D:/Research/Fire-Detection/none-fire/3.jpg'

test_image=image.load_img(image_for_testing,target_size=(224,224))

test_image=image.img_to_array(test_image)

test_image=test_image/255

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

result=cnn.predict(test_image)

#result=cnn.predict_class(test_image)

Catagories=['fire','no fire']
```

```
image_show=PIL.Image.open(image_for_testing)

plt.imshow(image_show)

plt.title(Catagories[int(result[0][0]==0)])

plt.show()
```