



Fire Image Detection and Video Capturing Using Machine Learning (CNN) Algorithm and IoT Concepts

¹Subramanya R Karthik, ²Dr Leena Giri G, ³Dr Prabha R

¹M-Tech Student, ²Associate Professor, ³Professor

¹Department of Computer science and Engineering ,

¹Dr Ambedkar Institute of Technology, Bangalore, India

Abstract : Fire outbreak is one of the most frequent issues happening all over the world in the recent times. The destruction caused by these types of scenarios is generally very much tremendous towards Humans and Nature. In the recent days as observed from the local information received from the users of this system is found that the Live Fire Based Detection System have gained the reputation as compared to Conventional Sensor Based Fire Detection System.

Significant experiments have been conducted on Live Fire Detection Algorithm using Convolution Neural Network to achieve High Efficiency Fire Image Detection which is approximately up to 84% in detecting the fire with trained dataset.

Live Fire Based Detection System can be installed in the busy area like Commercial Mall, Residential Apartment, Multistoried complex/offices, National Highways and Busy Traffic Area.

IndexTerms – GSM Board SIM800A, Arduino UNO Board, LED Light and Buzzer..

I. INTRODUCTION

Fire is one of the major hazards endangering the human life, Economy, and local Environment. Due to this rapid increase in fire accidents, every building or passenger vehicle for public transportation are to be equipped with fire protection and fire prevention systems. These systems are consisting mainly of point-type thermal and smoke detectors which are needed to be installed in the proximity of the fire; otherwise, they may be easily failed without detecting the fire. These systems are having remarkable potential advantages over the traditional methods, such as a fast response and wide detection areas.

As observed in the recent days, in many places of common area of human environments namely city streets, industry, public transportation, Apartments, Commercial Malls, busy traffic roads, etc. are installed with the Camera with Closed-circuit television (CCTV) systems for surveillance purpose to monitor the untoward incidents. Nowadays CCTV is the main tool for the surveillance.

The existing infrastructure includes the video camera, communication network, graphic processing unit and monitor screen. Utilization of this system will reduce the purchase and installation cost and do not require additional products. The fire detection algorithm can be easily integrated in this infrastructure by installing the additional software when required.

Fire Detection plays major role for safety of people. To decrease destruction caused by fire, many fire detection systems are created. One can find many accurate solutions. Many of them are based on sensors, which is limited to indoors. If the Sensors are not working or configured properly it can cause heavy damage in case of real fire.

The smoke and fire caused by natural process which in turn produces the particles which can be easily detected by sensors when they are in closed vicinity to the fire. Because of swift developments in digital camera and Video Processing Techniques, there is a significant inclination of change in the conventional fire detection methods. To detect the Fire in large & open Areas, Video based Fire Detection Techniques are more appropriate one.

Without spending any cost, it is very much easy in developing the Video based Fire Detection System which can be used as Surveillance Camera. Instead of Standard Detection Methods, we can use this Video based Fire Detection System which provides several advantages over the latter stages in future.

Vision_x0002_based fire detection system responds quickly in context with any other conventional detection method since, it doesn't need any type of conditions to activate the device. By using this device wide range of area can be monitored.

This model consists of 3x3 sizes, 64 Convolution filters; Rectified Linear Activation unit (Re Lu), these functions are used in feature maps are also called as activation maps and this helps to reduce the dimensionality of the input data are to be device inputs. The next operation is called as the max pooling which selects the more elements from the feature map surrounded by filters.

Early detection of fire-accidents can save innumerable lives along with saving of properties from the permanent infrastructure damages and the consequent financial losses if any. In order to achieve high accuracy and robustness in dense urban areas, detection through local surveillance video stream is one of the most feasible and cost-effective solution, which is suitable for replacement of existing systems without the need of large infrastructure for installation or investment.

The Fire Images recorded by Surveillance videos form the data sets which are transformed into the frames. These dataset images are further classified as fire and non – fire images. Here the Fire images are 1000 & non – Fire Images are 1000. In these images, the dense layer constitutes the matrix vector multiplication. The Matrix values are trainable variables which can be reformed during back propagation.

II. LITERATURE SURVEY

[1] Title: Predictions and Detections of the Forest Fire based on the Deep Learning Approach

Author: S. Gayathri, P.V. Ajay Karthi, Sourav Sunil.

Abstract: Forest Fire is mainly caused due to human interference which in turn disrupts the human activity, Natural Climate. To Control these Destructions are caused from the Forest Fire, in this paper they proposed a method which uses the Image Processing Model and gray scaling for detection of the forest fire using the deep learning based algorithms.

[2] Title: Additive Neural Network for Forest Fire Detection.

Author: Hongyi pan, Diaa badawi, Xi Zhang & Ahmet enis centin

Abstract: In the above, they have introduced the Additive Deep Neural Network which was computationally more efficient and base for the Wild Fire Detection Scheme by using videography.

[3] Title: Energy – Efficient Depp CNN for Smoke Detection in Foggy IoT Environment

Author: Salman Khan, Khan Muhammed, Shahid Mumtaz, Sung Wook Baik, Victor Hugo, CDE Albuquerque.

Abstract: In the above paper, the author have proposed the energy efficient system for detecting smoke in both Foggy and Normal IOT Environment conditions in the early stages which is Mainly based on Deep CNN. The above method uses the VGG – 16 Architecture.

[4] Title: Image Segmentation Using Deep Learning

Author: Shervin Minae, Yuri Boykov, Fatich Porikli, Antonio Plaza, Nasser Kehtarnavaz, Demetri Terzopoulous

Abstract: In the Above Paper, Authors have done the Survey on the recent literatures which covers the Semantic and Instance Segmentation this includes the Convolution Pixel – Labeling Network Visual Attention Models, They have also done a study on Strengths, Challenges and Relationships of the above Deep Learning Based Segmentation Models.

III. THE PROPOSED METHOD

Few years back, most of the researches on fire detections were focused on Conventional feature extraction methods. The drawback of this procedure is the time-consuming process and low performance in detecting fire. These methods are particularly in varying lightings; surveillance with shadows and fire-colored objects tends to create the false alarms.

CONVOLUTIONAL NEURAL NETWORK:

The working process of visual perception of the living species is the inspiration for CNN which is a deep learning framework. Since the launch of first well-known DL architecture Le-Net for hand-written digits classification, it has shown promising results for combating different problems including action recognition, image classification, scene labeling, object localization, visual saliency detection, object tracking, image segmentation, indexing and retrieval and speech processing.

In this proposed method, the classification of architecture is a combination of convolution and max – pooling which is a conventional process of convolution neural networking. However, to obtain the fast classification, a small network has been selected, which shows nine layers of CNN. Here is an image with RGB color, subsequently goes through convolutional operations with kernel of size 3x3. For the layer three also the same structure is applied. For convolutional layer two and five, Max pooling 3x3 with stride 2 has been applied. There are 16 feature maps in one to four layers. The layers of five and six have only one feature map. The layer seven and eight are fully connected. The last one is fully connected layer's output is given as input to a 3-way Soft max function which produces the distribution over 3 class labels. In comparison with hand-engineered feature based procedures are in most of the application domains, CNN have been used extensively in the image classification to gain more classification accuracy over the large scale of datasets. To generate the feature maps in convolution operations, many kernels with various sizes are used in the input data This way the generated feature maps will be input to the subsequent operation called as subsampling or pooling, in which most of the activations are selected within the small neighborhood. These operations will play an important role in reducing the feature vectors dimension and to achieve the translation invariance to certain degree.

Fully connected layer is one of the main layers of the CNN pipeline, where high-level abstractions are modeled from the input data and will be connected fully and this convolution layers contain neurons, whose weights learnt and adjusted as required, so that input data can be represented better during training process.

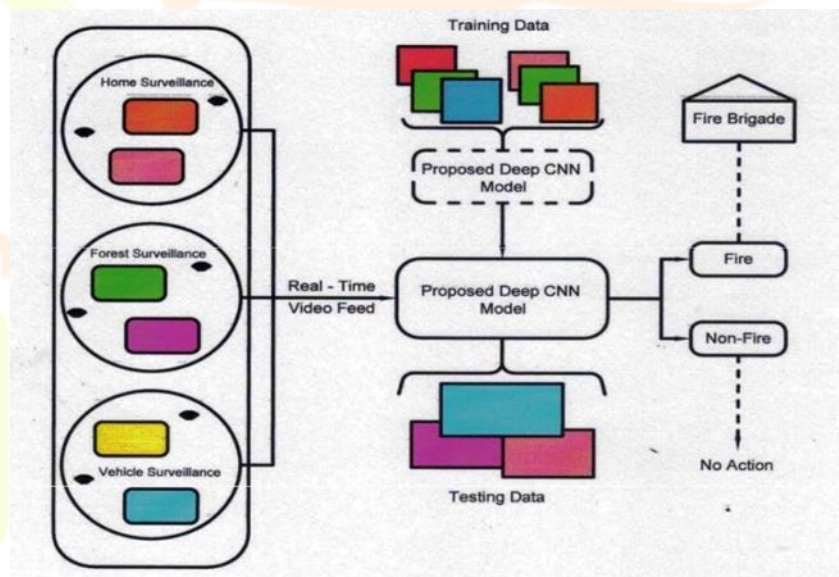
FIRE DETECTION IN SURVEILLANCE VIDEOS USING CNN:

Deep learning architectures will automatically learn deep features from raw data. This is the known fact that which was agreed by the research community. To generate the optimal solution for the intended problem of some extra work is required to train the different models with the various settings. For the same reason numerous models with different parameter settings were trained based on quality of gathered data and nature of the problem. Transfer learning strategy is one of the techniques used here, where the solutions for complicated problems can be addressed by applying the knowledge from previous learnt. The best architecture was finalized after working on standard datasets, which had the capability to detect fire with most accuracy, both in outdoor and indoor surveillance videos. To obtain expected output, test image is fed as input. The result is probabilities for both the sections are fire and non – fire. The final label for the test image is taken as the maximum probability score between fire and non – fire sections.

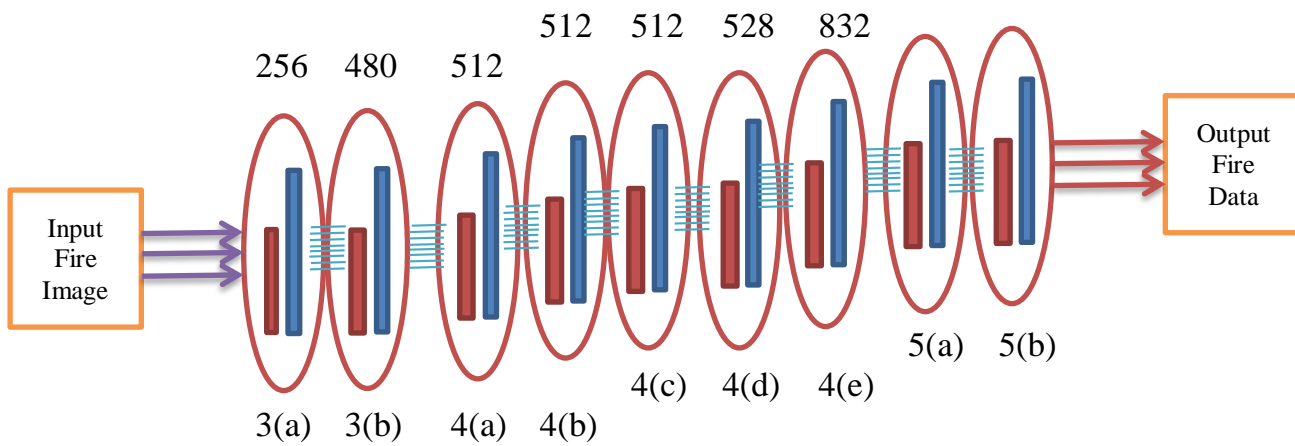
CONVOLUTIONAL NEURAL NETWORK ARCHITECTURES:

The flow of image fire detection algorithm is designed by applying CNN, which has region proposal function, feature extraction function and feature classification function. First and foremost, image is taken as input by CNN and region proposals by convolution and pooling is generated as output. Next, the presence or absence of fire in proposal regions were decided by the region-based object detection CNN along fully connected layers, convolutional layers and pooling layers. The convolutional layer is quite different from other neural networks and it is a main part of CNN, which makes use of connection weights and weighted sums to produce feature maps from original images using transform filters known as convolution kernel.

The next function is called as subsampling or pooling which takes these feature maps as input where the most of actuators are chosen within the small neighborhood. These operations are the key to minimize the dimension of feature vector and to accomplish the translation invariance to some extent. Next layer of the CNN is fully connected layer, which is also one of the very important layers; thereby the input data of high – level conceptions will be modeled. The fully connected and convolution layers contain neurons. Where weights of these neurons have are to be learnt and calibrated for the better characterization of the input data during training procedure.



Surveillance CNN Data Model

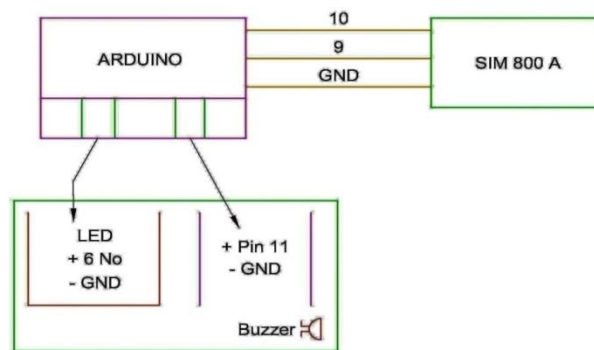


CONVOLUTION POOLING MODEL

In convolution operation, the several kernels of different sizes are applied on the input data to generate feature maps. These features maps are input to the next operation is known as subsampling or pooling, where the maximum activations are selected from them within the small neighborhood. These operations are important for reducing the dimension of feature vectors and achieving translation invariance up to the certain degree. Another important layer of the CNN Pipeline is fully connected layer, where high – level abstractions are modeled from the input data. Among these three main operations, the convolution and fully connected layers contain neurons whose weights are learnt and adjusted for better representation of the input data during training process.

IoT SETUP EXPLANATION:

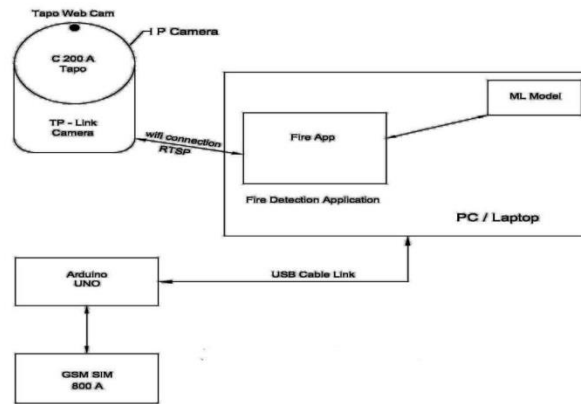
In the figure, the GSM Board is used to Make Call and Send SMS When Fire image is detected. The Interface takes place between Arduino UNO and GSM SIM800A Board by connecting Pin no 10, 9 and GND (Ground Pin) from Arduino UNO Board to the SIM800A GSM Board. In Arduino Board, LED light pin is connected from pin no 6 to GND pin (Attached with Jump wire). Buzzer pin is connected from pin no 11 to GND pin (Small Pin). The Overall Connection is used by USB cable from PC Source.



IoT Setup Configuration

BLOCK DIAGRAM EXPLANATION:

In the figure, TP Camera (Tapo C200A) is taken as an IP Camera where it is connected to Fire dataset application which is inherited Fire Detection ML Model in the PC device. In TP Link Camera the IP Address is auto-generated by RTSP (Real Time Streaming Profile) Address, which requires TP User Account and Password which is written in ML dataset code for execution. Through IoT Setup Device, The dataset is implemented and executed by USB Cable connection from PC to IoT Device.



Block Diagram for the Project

IV IMPLEMENTATION AND RESULTS

In this project work, ML (CNN Model) and Python is used as development language. The dataset is taken and trained from Kaggle website. ML Model is created and used for testing the fire detection. Once, the fire is detected, the alarm (sound) will be ON & LED will be blinked. If 5 Times the fire is detected then make a Call to Concern Person By sending SMS and E – Mail to Concern Persons. Using Random Fire Video the Fire Image is detected Since, We cannot create natural Massive Fire On our Own. The Candle Light or Lamp Fire Light or Match Light cannot be detected with this setup (Not Considered as Fire to Alarm).

Objectives:

Detect - Massive Fire using ML Model Dataset.

Alert the Concerned People through SMS, Buzzer Alarm, LED Light, GSM Core, Gmail Address from Sender to Receiver Module.

Working of the Project:

Camera Captures the images [Fire] and then sends to the application [Fire Detection or Smoke Application].

The Application will send the image to the ML Model to detect whether the image contains Fire & Smoke.

If Detected, It will produce Alarm & LED Blink.

If it is repeated 5 Times then the application will make a Call Send SMS, Send Email with Fire Image to the concern person.

Fire Smoke Detection Dataset from Kaggle:

A Huge Number of Data is required to Frame the Algorithms which are based on CNN. These needs cannot be met by Current Small Scale - Image or Databases for the current Project around 3000 images are gathered Video/Fire from Small/Large Public Fire. Three Types of Information which are general information, Background Information and Fire Information are provided by each annotation file of the image. Along with this for the finding of Fire & Smoke (More Number) & Further Research is required in this direction.

```

In [1]: import tensorflow as tf
import keras.preprocessing
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
TRAINING_DIR = "C:\Users\SRK\Downloads\FIRE-SMOKE-DATASET\Train"
training_datagen = ImageDataGenerator(rescale = 1./255,
                                      horizontal_flip=True,
                                      rotation_range=30,
                                      height_shift_range=0.2,
                                      fill_mode='nearest')

VALIDATION_DIR = "C:\Users\SRK\Downloads\FIRE-SMOKE-DATASET\Test"
validation_datagen = ImageDataGenerator(rescale = 1./255)

train_generator = training_datagen.flow_from_directory(TRAINING_DIR,
                                                      target_size=(224, 224),
                                                      class_mode='categorical',
                                                      batch_size = 64)

validation_generator = validation_datagen.flow_from_directory(
    VALIDATION_DIR,
    target_size=(224, 224),
    class_mode='categorical',
    batch_size= 16)

Found 2700 images belonging to 3 classes.
Found 300 images belonging to 3 classes.

In [4]: from tensorflow.keras.optimizers import Adam
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(96, (3,3), strides=(4,4), activation='relu', input_shape=(224, 224, 3)), tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
    tf.keras.layers.Conv2D(256, (5,5), activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size = (3,3), strides=(2,2)),
    tf.keras.layers.Conv2D(384, (5,5), activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size = (3,3), strides=(2,2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(5, activation='softmax')])
model.compile(loss='categorical_crossentropy',
              optimizer=Adam(lr=0.0005),
              metrics=['acc'])
history = model.fit(
    train_generator,

```

IMAGE DATA COLLECTION

```

In [2]: from tensorflow.keras.optimizers import Adam
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(96, (3,3), strides=(4,4), activation='relu', input_shape=(224, 224, 3)), tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
    tf.keras.layers.Conv2D(256, (5,5), activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size = (3,3), strides=(2,2)),
    tf.keras.layers.Conv2D(384, (5,5), activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size = (3,3), strides=(2,2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dense(5, activation='softmax')])
model.compile(loss='categorical_crossentropy',
              optimizer=Adam(lr=0.0005),
              metrics=['acc'])
history = model.fit(
    train_generator,
    steps_per_epoch = 5,
    epochs = 5,
    validation_data = validation_generator,
    validation_steps = 15
)

C:\Users\SRK\Anaconda3\lib\site-packages\tensorflow\python\keras\optimizer_v2\optimizer_v2.py:374: UserWarning: The 'lr' argument is deprecated, use 'learning_rate' instead.
  warnings.warn()

Epoch 1/5
5/5 [#####] 2s/step - loss: 1.0860 - acc: 0.3938 - val_loss: 1.0495 - val_acc: 0.4042
Epoch 2/5
5/5 [#####] 2s/step - loss: 1.4393 - acc: 0.4989 - val_loss: 0.9121 - val_acc: 0.5833
Epoch 3/5
5/5 [#####] 2s/step - loss: 0.8762 - acc: 0.5844 - val_loss: 0.7955 - val_acc: 0.6375
Epoch 4/5
5/5 [#####] 2s/step - loss: 0.8238 - acc: 0.6281 - val_loss: 0.7518 - val_acc: 0.6917
Epoch 5/5
5/5 [#####] 2s/step - loss: 0.8058 - acc: 0.6156 - val_loss: 0.8013 - val_acc: 0.6500

In [5]: model.save('fire_smoke_detector')
INFO:tensorflow:Assets written to: fire_smoke_detector/assets

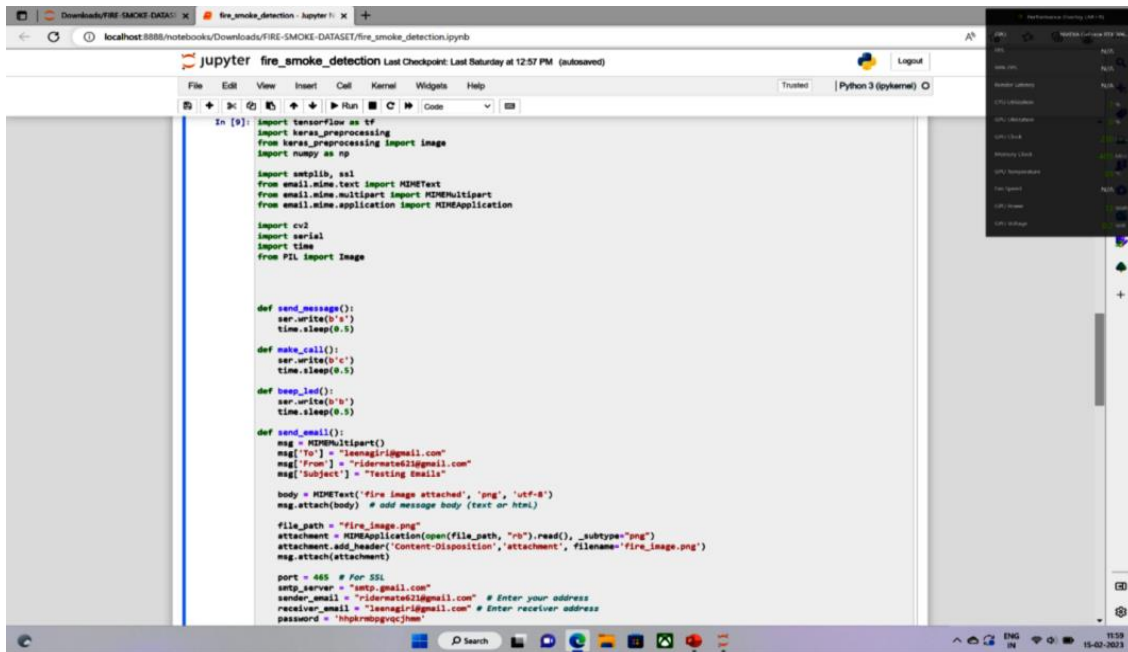
In [6]: detector = tf.keras.models.load_model('fire_smoke_detector')

In [7]: import numpy as np

```

DATASET IMPLEMENTATION USING CNN

Research Through Innovation



```

In [9]: import tensorflow as tf
import keras.preprocessing
from keras.preprocessing import image
import numpy as np

import cv2
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.application import MIMEApplication

import cv2
import serial
import time
from PIL import Image

def send_message():
    ser.write(b'b')
    time.sleep(0.5)

def make_call():
    ser.write(b'a')
    time.sleep(0.5)

def beep_led():
    ser.write(b'b')
    time.sleep(0.5)

def send_email():
    msg = MIMEMultipart()
    msg['To'] = "jaanagiri@gmail.com"
    msg['From'] = "riderate2@gmail.com"
    msg['Subject'] = "Testing Emails"

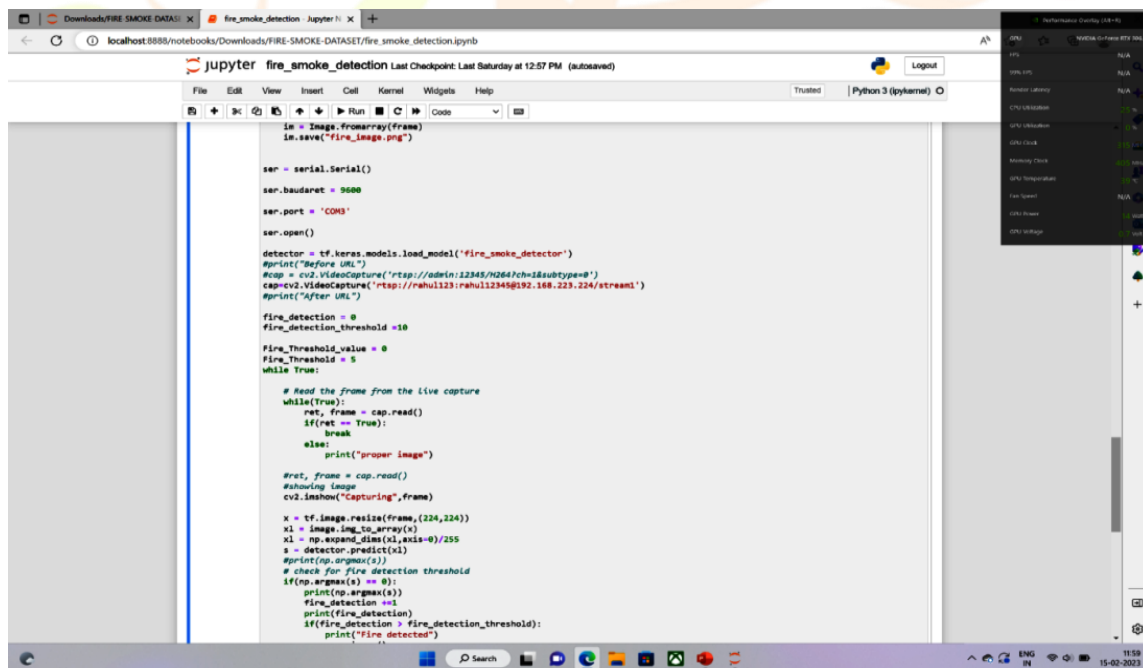
    body = MIMEText('fire image attached', 'png', 'utf-8')
    msg.attach(body) # add message body (text or html)

    file_path = "fire_image.png"
    attachment = MIMEApplication(open(file_path, "rb").read(), _subtype="png")
    attachment.add_header('Content-Disposition', 'attachment', filename='fire_image.png')
    msg.attach(attachment)

    port = 465 # For SSL
    smtp_server = "smtp.gmail.com"
    sender_email = "riderate2@gmail.com" # Enter your address
    receiver_email = "jaanagiri@gmail.com" # Enter receiver address
    password = 'hpkrrbpgvcjham'

```

SENDING THE FIRE IMAGE DATA FROM SENDER TO RECIEVER EMAIL ADDRESS THROUGH TP LINK CAMERA



```

img = image.fromarray(frame)
img.save("fire_image.png")

ser = serial.Serial()
ser.baudrate = 9600
ser.port = 'COM3'
ser.open()

detector = tf.keras.models.load_model("fire_smoke_detector")
#print("Before URL")
#cap = cv2.VideoCapture("rtsp://admin:12345@192.168.223.224/stream1")
cap=cv2.VideoCapture("rtsp://rahu123:rahu12345@192.168.223.224/stream1")
#print("After URL")

fire_detection = 0
fire_detection_threshold = 10

Fire_Threshold_value = 0
Fire_Threshold = 5
while True:

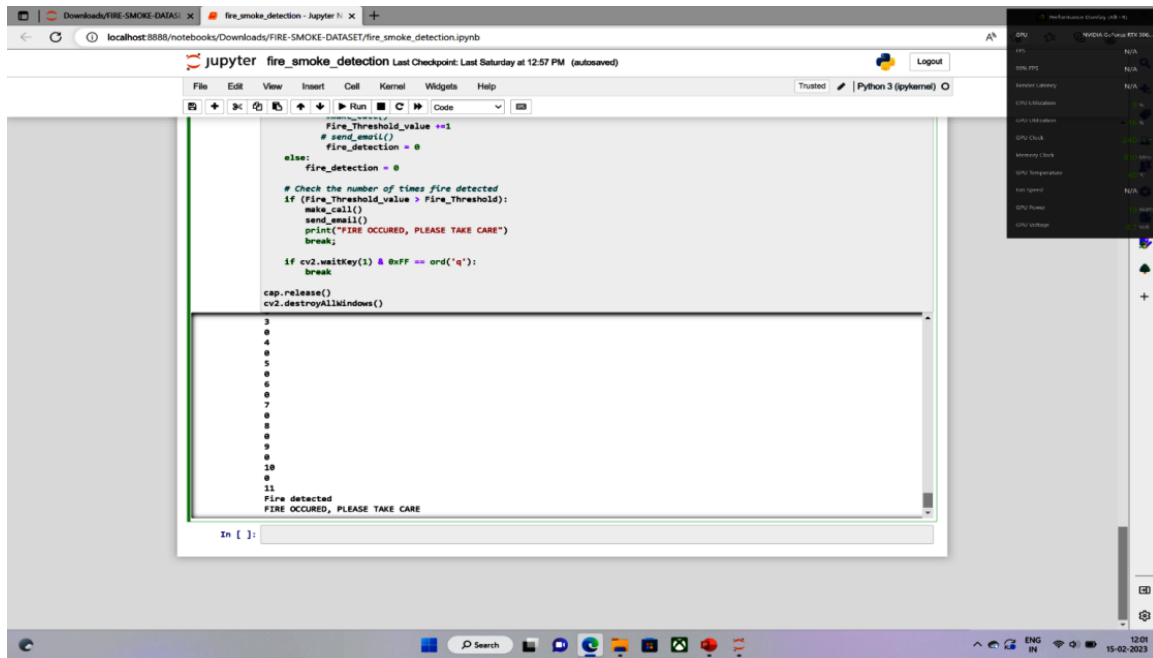
    # Read the frame from the live capture
    while(True):
        ret, frame = cap.read()
        if(ret == True):
            break
        else:
            print("proper image")

    #ret, frame = cap.read()
    #showing Image
    cv2.imshow("Capturing", frame)

    x = tf.image.resize(frame, (224,224))
    x1 = image.img_to_array(x)
    x1 = np.expand_dims(x1,axis=0)/255
    s = detector.predict(x1)
    #print(np.argmax(s))
    # check for fire detection threshold
    if(np.argmax(s) == 0):
        print(np.argmax(s))
        fire_detection +=1
        print(fire_detection)
        if(fire_detection > fire_detection_threshold):
            print("fire detected")

```

WRITING THE IP ADDRESS OF VIDEO CAPTURING FOR FIRE IMAGE CAPTURE



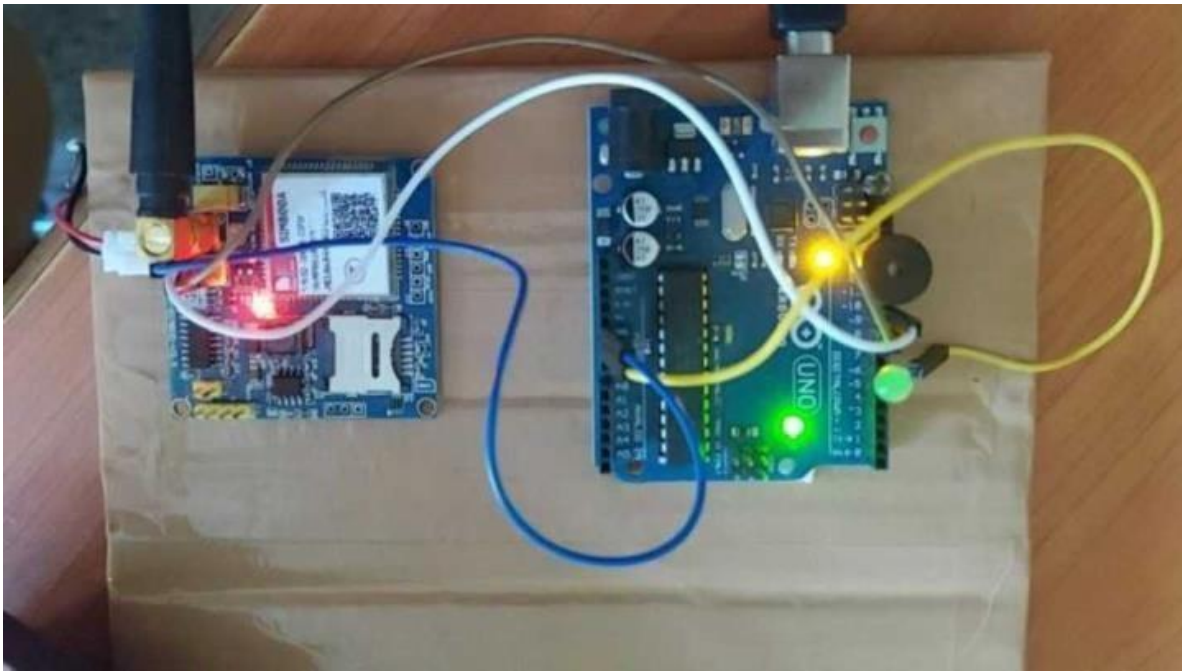
```
fire_threshold_value += 1
# send_email()
fire_detection = 0
else:
    fire_detection = 0
# Check the number of times fire detected
if (fire_threshold_value > Fire_Threshold):
    make_call()
    send_email()
    print("FIRE OCCURED, PLEASE TAKE CARE")
    break;
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
cap.release()
cv2.destroyAllWindows()
```

```
3
4
5
6
7
8
9
10
11
Fire detected
FIRE OCCURED, PLEASE TAKE CARE
```

AFTER 6 TIMES OF FIRE CAPTURING WE GET FIRE DETECTED WITH ALERT MESSAGE



FIRE IMAGE DETECTED AND SENT FROM SENDER GMAIL TO RECIEVERS GMAIL



SIGNALLING OF FIRE DETECTION USING CNN AND IOT

1) Neutral [Neither Fire or Smoke Image]	1 [Threshold Value]
2) Smoke Image	2 [Threshold Value]
3) Fire Image	0 [Threshold Value]

RESULT OF THE PROJECT

V CONCLUSION

Nowadays, high processing abilities of smart devices have shown encouraging results (Outputs) in surveillance systems to identify different abnormal events like fire and other emergency as it is known fire is one of the destructive events which can result in great loss if not attended on them. For the same reason, developing early fire detection is very important. Therefore, here we are proposing a CNN Architecture for fire detection for surveillance videos which are cost – effective. In spite of the fact the current work has improved the accuracy of the fire detection; more number of false alarms still exists further research is required in this direction, along with this for the detection of smoke and fire, the existing fire detection technology can be tuned further. In this way, more complicated situation in the real – world can be handled more efficient way through video – surveillance system. The fire data is sent through the email address thus fire image is detected by TP - Camera.

REFERENCES

- (1) K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H. H. Ge Wang, and S. W. Baik, "Secure Surveillance Framework for IoT Systems Using Probabilistic Image Encryption," *IEEE Trans. Ind. Inform.*, doi: <https://doi.org/10.1109/TII.2018.2791944> (Vol : 14, no ! 8, pp. 3679 - 3689 August 2018).
- (2) K. Muhammad, J. Ahmad, and S. W. Baik, "Early fire detection using convolutional neural networks during surveillance for effective disaster management," *Neurocomputing*, [Online]. <https://www.sciencedirect.com/science/article/pii/S0925231217319203> vol. 288, pp. 30-42, May 2018.
- (3) J. Choi and J. Y. Choi, "An integrated framework for 24-hours fire detection," in *Proc. Eur. v Conf. Comput. Vis.*, ol 28, pp. 463-479, June 2016.
- (4) H. J. G. Haynes. (2016). *Fire Loss in the United States During 2015*. [Online]. Available: <http://www.nfpa.org/> vol 27, pp.289-389, September 2016.
- (5) C.-B. Liu and N. Ahuja, "Video based fire detection," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, vol 15, pp. 134-137, Aug 2004.
- (6) T.-H. Chen, P.-H. Wu, and Y.-C. Chiou, "An early fire-detection method based on image processing," in *Proc. Int. Conf. Image Process. (ICIP)*, vol 24, pp. 1707-1710, Oct 2004.
- (7) B. U. Toreyin, Y. Dedeoglu, U. Gudukbay, and A. E. İetin, "Computer vision based method for real-time fire and fire detection," *Pattern Recognition. Lett.*, vol. 27, pp. 49-58, Jan. 2006.
- (8) J. Choi and J. Y. Choi, "Patch-based fire detection with online outlier learning," in *Proc. 12th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, vol 28, pp. 1-6, Aug. 2015.
- (9) T. İelik and H. Demirel, "Fire detection in video sequences using a generic color model," *Fire Safety J.*, vol. 44, no. 2, pp. 147-158, December 2009.
- (10) P. V. K. Borges and E. Izquierdo, "A Probabilistic approach for vision based fire detection in videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 5, pp. 721-731, May 2010.

