FIRE DETECTION AND ALARMING SYSTEM

Major Project Report

Submitted in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Technology (B. Tech)

in

COMPUTER SCIENCE AND ENGINEERING

By

P SRI SAI KEERTHANA	**18AG1A05G0**
P BHANU PRAKASH REDDY	**18AG1A05G1**
M PRAVEEN	**18AG1A05F4**
T R SAI SRUTHI	**18AG1A05H6**

Under the Esteemed Guidance of

Mr. SHASHANK TIWARI

Assistant Professor

Department of Computer Science and Engineering

ACE ENGINEERING COLLEGE

An Autonomous Institution

(NBA ACCREDITED B. TECH COURSES: EEE, ECE, MECH, CIVIL & CSE, ACCORDED NAAC 'A' GRADE)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)

Ghatkesar, Hyderabad - 501 301

JUNE 2022

FIRE DETECTION AND ALARMING SYSTEM

Major Project Report

Submitted in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Technology (B. Tech)

in

COMPUTER SCIENCE AND ENGINEERING

By

P SRI SAI KEERTHANA	**18AG1A05G0**
P BHANU PRAKASH REDDY	**18AG1A05G1**
M PRAVEEN	**18AG1A05F4**
T R SAI SRUTHI	**18AG1A05H6**

Under the Esteemed Guidance of

Mr. SHASHANK TIWARI

Assistant Professor

Department of Computer Science and Engineering

ACE ENGINEERING COLLEGE

An Autonomous Institution

(NBA ACCREDITED B. TECH COURSES: EEE, ECE, MECH, CIVIL & CSE, ACCORDED NAAC 'A' GRADE)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)
Ghatkesar, Hyderabad - 501 301

JUNE 2022

ACE Engineering College

An Autonomous Institution

(NBA ACCREDITED B. TECH COURSES: EEE, ECE, MECH, CIVIL & CSE, ACCORDED NAAC 'A' GRADE)

Ghatkesar, Hyderabad- 501 301

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

Website: www.aceec.ac.in E-mail: info@aceec.ac.in

CERTIFICATE

This is to certify that the Major project work entitled **"FIRE DETECTION AND ALARMING SYSTEM "** is being submitted by **P SRI SAI KEERTHANA (18AG1A05G0), P BHANU PRAKASH REDDY (18AG1A05G1), M PRAVEEN (18AG1A05F4), T R SAI SRUTHI (18AG1A05H6),** in partial fulfillment for the award of Degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** to the Jawaharlal Nehru Technological University, Hyderabad during the academic year 2021-22 is a record of bonafide work carried out by him/her under our guidance and supervision.

The results embodied in this report have not been submitted by the student to any other University or Institution for the award of any degree or diploma.

Internal Guide

Mr. Shashank Tiwari

Assistant Professor

Dept. of CSE

Head of the Department

Dr.M.V.Vijay Saradhi

Professor and Head

Dept. of CSE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I would like to express my gratitude to all the people behind the screen who have helped me transform an idea into a real time application.

I would like to express my heart-felt gratitude to my parents without whom I would not have been privileged to achieve and fulfill my dreams.

A special thanks to our Secretary, **Prof. Y. V. GOPALA KRISHNA MURTHY,** for having founded such an esteemed institution. I am also grateful to our beloved principal, **Dr. B. L. RAJU** for permitting us to carry out this project.

I profoundly thank **Prof. Dr.M.V.Vijay Saradhi**, Head of the Department of Computer Science and Engineering, who has been an excellent guide and also a great source of inspiration to my work.

I extremely thank **Mrs. Soppari Kavitha,** Associate Professor, Project coordinator, who helped us in all the way in fulfilling of all aspects in completion of our Major-Project.

I am very thankful to my internal guide **Mr. Shashank Tiwari,** Assistant Professor, who has been an excellent and also given continuous support for the completion of my project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the people mentioned above who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, I would like to thank all the other staff members, both teaching and non-teaching, who have extended their timely help and eased my task.

P SRI SAI KEERTHANA (18AG1A05G0)
P BHANU PRAKASH REDDY (18AG1A05G1)
M PRAVEEN (18AG1A05F4)
T R SAI SRUTHI (18AG1A05H6)

DECLARATION

This is to certify that the work reported in the present project titled **"FIRE DETECTION AND ALARMING SYSTEM"** is a record of work done by us in the department of Computer Science and Engineering, ACE Engineering College.

No part of the thesis is copied from books/journals/internet and whenever the portion is taken, the same has been duly referred to in the text; the reports are based on the project work done entirely by us not copied from any other source.

P SRI SAI KEERTHANA (18AG1A05G0)
P BHANU PRAKASH REDDY (18AG1A05G1)
M PRAVEEN (18AG1A05F4)
T R SAI SRUTHI (18AG1A05H6)

ABSTRACT

History has proven that early detection of a fire and the signaling of an appropriate alarm remain significant factors in preventing large losses due to fire. Properly installed and maintained fire detection and alarm systems can help to increase the survivability of occupants and emergency responders while decreasing property losses.

The technologies underlying fire and smoke detection systems play a crucial role in ensuring and delivering optimal performance in modern surveillance environments. In fact, fire can cause significant damage to lives and properties.

Considering that the majority of cities have already installed camera-monitoring systems, this encouraged us to take advantage of the availability of these systems to develop cost-effective vision detection methods.

From sprawling urban to dense jungles, fire accidents pose a major threat to the world. These could be prevented by deploying fire detection systems, but the prohibitive cost, false alarms, need for dedicated infrastructure, and the overall lack of robustness of the present hardware and software-based detection systems have served as roadblocks in this direction. In this work, we endeavor to make a stride towards detection of fire in videos using Deep learning. Deep learning is an emerging concept based on artificial neural networks and has achieved exceptional results in various fields including computer vision. We plan to overcome the shortcomings of the present systems and provide an accurate and precise system to detect fires as early as possible and capable of working in various environments thereby saving innumerable lives and resources.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Fire accidents pose a serious threat to industries, crowded events, social gatherings, and densely populated areas that are observed across India. These kinds of incidents may cause damage to property, the environment, and pose a threat to human and animal life.

According to the recent National Risk Survey Report, Fire stood at the third position overtaking corruption, terrorism, and insurgency thus posing a significant risk to our country's economy and citizens.

The recent forest fires in Australia reminded the world, of the destructive capability of fire and the impending ecological disaster, by claiming millions of lives and resulting in billions of dollars in damage.

Early detection of fire accidents can save innumerable lives along with saving properties from permanent infrastructure damage and the consequent financial losses. In order to achieve high accuracy and robustness in dense urban areas, detection through local surveillance is necessary and also effective.

Traditional optoelectronics fire detection systems have major disadvantages: The requirement of separate and often redundant systems, fault-prone hardware systems, regular maintenance, false alarms, and so on. Usage of sensors in hot, dusty industrial conditions is also not possible. Thus, detecting fires through surveillance video stream is one of the most feasible, cost-effective solutions suitable for the replacement of existing systems without the need for large infrastructure installation or investment.

The existing video-based machine learning models rely heavily on domain knowledge and feature engineering to achieve detection therefore, have to be updated to meet new threats. We aim to develop a classification model using Deep learning and Transfer Learning to recognize fires in images/video frames, thus ensuring early detection and save manual work. This model can be used to detect fires in surveillance videos. Unlike existing systems, this neither requires special infrastructure for setup like hardware-based solutions, nor does it need domain knowledge and prohibitive computation for development.

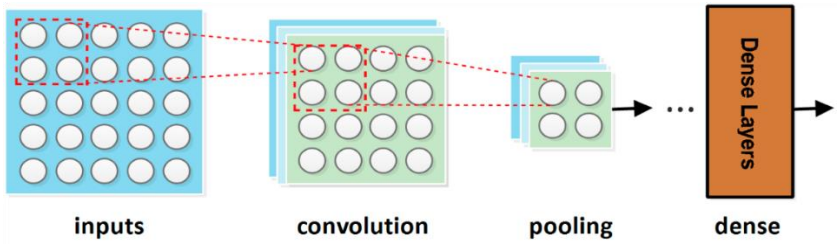


Figure 1. General Representation of CNN Model.

2.LITERATURE SURVEY

Among the different computer-based approach detecting fire, the prominent approaches we found were using Artificial Neural network, Deep Learning, Transfer learning and convolutional neural network. Artificial Neural Network based approaches seen in paper [2] uses Levenberg Maraquardt training algorithm for a fast solution. The accuracy of the algorithm altered between 61% to 92%. False positives ranged from 8% to 51%. This approach yielded high accuracy and low false positive rate, yet it requires immense domain knowledge.

In this paper [3], The author says that the present hardware-based detection systems offer low accuracy along with high occurrence of false alarms consequently making it more likely to misclassify actual fires. It is also not suitable for detecting fires breaking out in large areas such as forests, warehouses, fields, buildings or oil reservoirs.

The authors used a simplified YOLO (You Only Look Once) model with 12 layers. Image augmentation techniques such as rotation, adjusting contrast, zooming in/out, saturation and aspect ratio were used to create multiple samples of each image, forming 1720 samples in total. It aims to draw a bounding box around the flame region. It outperformed existing models when the color features of the flames varied from those in training set. Paper [4] provides two approaches First approach is to perform training on the data set using Transfer Learning and later fine tune it. The next approach was to extract flame features, fuse them and classify it using a machine learning classifier. The transfer learning algorithms used were Xception, Inception V3, ResNet-50, trained in ImageNet. In the first approach, accuracy up to 96% was achieved. The second approach, stacking Xgboost and lightbgm achieved an AUC of 0.996. Transfer learning models greatly reduces the training time required for our model. It requires comparatively smaller data set. Both approaches don't require any sort of domain knowledge.

In works [7] and [8], Deep CNN approach was taken to detection and localization of fires. The accuracy obtained was between 90 to 97% in both of these papers. This approach is time

consuming and training was performed using Nvidia GTX Titan X with 12 GB of onboard memory.

Traditional Machine learning using feature extraction yielded high accuracy and low false positive rate, yet it requires immense domain knowledge i.e., about color model, color-space, patterns and motion vectors of flames. when the object changes, the models need to be rebuilt for the new objects.

The traditional approach to feature engineering [12] is manual in nature. It involves handcrafting features incrementally using domain knowledge, a tedious, time consuming, and error-prone process. The resultant model is problem dependent and might not perform well on new data.

Automated feature engineering ([3][5]) improves upon this inefficient workflow by automatically extracting useful and meaningful features from data with a framework that can be applied to any problem. It not only cuts down on the time spent, but creates features that can be interpreted and prevents data leakage.

With transfer learning, instead of developing a model from scratch, we can start from a pre-trained model with necessary fine-tunings. These models can be imported directly from Keras. The use of pre-trained models saves a lot of computational work, which otherwise, would require high end GPU's. Inception V3, Inception-ResNet-V2 were found to be ideal algorithms for feature extraction as they showed promising results with high accuracy ([3]).

With transfer learning, instead of developing a model from scratch, we can start from a pre-trained model with necessary fine-tunings. These models can be imported directly from Keras. The use of pre-trained models saves a lot of computational work, which otherwise, would require high end GPU's. Inception V3 is found to be ideal algorithms for feature extraction as they showed promising results with high accuracy.

3.SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

3.1.1 Janku.p

Convolutional neural networks are a special kind of artificial neural network that can mimic human brain activity to analyze data with supervised learning. CNN is a modified multilayer perceptron, which means a fully connected network. It consists of several layers namely, the input layer, output layer, and many hidden layers to make it happen. These hidden layers are convolutional hence the name convolutional neural networks. It offers beyond the limit abilities toper form object detection. These convolutional layers use several mathematical models to critically evaluate and analyze data. Then these outputs of the previous layers are passed to the next layers. There is a chance of overfitting since the network is fully connected. To avoid this situation, CNN exploits the hierarchical pattern in the data and sorts them according to their complexity from simpler to complex patterns engraved in the layers. The input is given as a tensor with several inputs x height x width x channels of input. Now the image is in an abstract form, then the layers convert this abstract image into a feature map. This is repeated layer after layer which simulates the working of brain neurons. Since it is a fully connected network all the output gets filtered and combined as a single output in the output layer. The number of filters is directly proportional to the feature map size. Along with the convolutional layer, it also includes a pooling layer to reduce the size of the output from the convolutional layers.

3.1.2 Shen.D

YOLO is an abbreviation for the term ‘You Only Look Once’. This is an algorithm th**at** detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images. YOLO algorithm employs convolutional neural networks (CNN) to detect objects in real-time. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects. This means that prediction in the entire image is done in a single algorithm run. The CNN is used to predict various class probabilities and bounding boxes simultaneously.

The YOLO algorithm consists of various variants. Some of the common ones include tiny YOLO and YOLOv3.

3.1.3 Li

Transfer learning is a machine learning method where we reuse a pre-trained model as the starting point for a model on a new task. The transfer learning model used in his paper were Xception learning which is similar to the inception model. Another algorithm used is a stacking algorithm. transfer learning is a machine learning method where we reuse a pre-trained model as the starting point for a model on a new task. Here he stacked Xgboost and lightbgm algorithms. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks**.** LightGBM is a gradient boosting framework based on decision trees to increase the efficiency of the model and reduces memory usage.

3.1.4 K. Muhammad

Inception-ResNet-v2 is a convolutional neural architecture that builds on the Inception family of architectures but incorporates residual connections (replacing the filter concatenation stage of the Inception architecture). The network of the pre trained model is 164 layers deep and can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images.

3.2. PROPOSED SYSTEM

- To create a real-time fire detection and alarming system that produces an alarm upon detection of fire by examining the surveillance video. The goal of this project is to create an algorithm that allows the detection of fire that can be accurate. This can then be used to trigger the alarm for the indication of fire.
- The approach used here is a transfer learning model called inceptionNet V3 with Adam optimizer for the classification purpose producing a faster and more accurate result.
- In order to check the real-time working of the model we are using OpenCV and by using the CNN model built the prediction is acquired and if the fire is detected we are introducing an alarming system that sends a message to the owner regarding the fire.

ADVANTAGES:

- Speed and accuracy
- Requires less data and faster training process.

3.3 SOFTWARE REQUIREMENT SPECIFICATION

3.3.1.OVERALL DESCRIPTION

The software requirements specification is a complete specification and description of the requirements of the software that needs to be fulfilled for the successful development of the software system. System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.
- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements)
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization.
- The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for addingnew modules and debugging old running system.
- All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminaryinvestigation.

3.3.2. SOFTWARE REQUIREMENTS

- Operating System : Windows 7/8(or Higher)/ MAC/Linux
- Programming Language : Python
- Libraries : NumPy== 1.19.5,
  matplotlib == 3.5.1,
  TensorFlow==2.7.0

3.3.4. SOFTWARE INTERFACES

The required software is python latest version is recommended.

3.3.5. EXTERNAL INTERFACE REQUIREMENTSUSER INTERFACE

The user interface of this system is a user-friendly python Graphical User Interface.

3.3.6. OPERATING ENVIRONMENT

Windows XP/Linux/Mac

3.4. PROCESS MODEL USED WITH JUSTIFICATION

Machine learning life cycle is a standard which is used by software industry to develop good machine learning model.

STAGES IN MACHINE LEARNING LIFE CYCLE

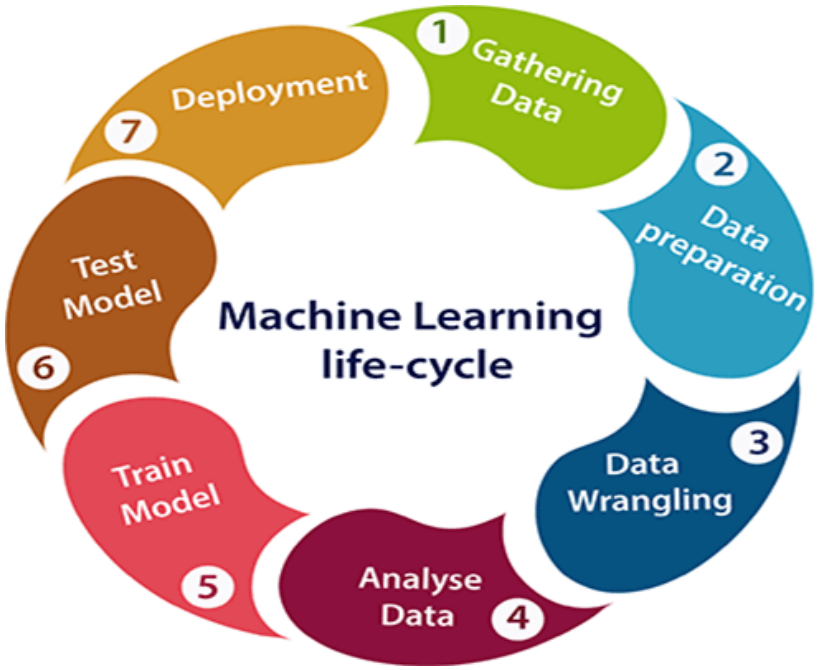


FIG 3.1 **MACHINE LEARNING LIFE CYCLE**

The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.

In the complete life cycle process, to solve a problem, we create a machine learning system called "model", and this model is created by providing "training". But to train a model, we need data, hence, life cycle starts by collecting data.

1. Gathering Data or Dataset:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most

important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

1. Identify various data sources
2. Collect data
3. Integrate the data obtained from different sources

By performing the above task, we get a coherent set of data, also called as a dataset. It will be used in further steps.

2. Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

1. **Data exploration:** It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data. A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.
2. **Data pre-processing:** Now the next step is pre processing of data for its analysis. Pre processing is a phase in which we clean the dataset and make it ready for making it more useful to solve the problem by extracting knowledge from it.

3. Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful.

In real-world applications, collected data may have various issues, including:

1. Missing Values
2. Duplicate data
3. Invalid data
4. Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

1. Selection of analytical techniques
2. Building models
3. Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such

as Classification**,** Regression**,** Cluster analysis**,** Association, etc. then build the model using prepared data, and evaluate the model.

Hence, in this step, we take the data and use machine learning algorithms to build the model.

5. Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

6. Test Model

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

7. Performance Metrices

After doing the usual Feature Engineering, Selection, and of course, implementing a model and getting some output in forms of a probability or a class, the next step is to find out how effective is the model based on some metric using test datasets. Different performance metrics are used to evaluate different Machine Learning Algorithms.

Eg. for classification problems we check for accuracy of the model and we can estimate its accuracy and performance by using accuracy score and confusion matrices.

8. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project

3.5.HARDWARE REQUIREMENTS SPECIFICATION

- Processor : Intel i3 (Min).
- Speed : 1.1Ghz
- RAM : 4gb (Min)
- Graphic card : 2 GB(Optional)
- Hard disk : 250GB (Min).
- Keyboard : Standard windows keyboard
- Mouse : Two or Three button mouse
- Monitor : SGVA

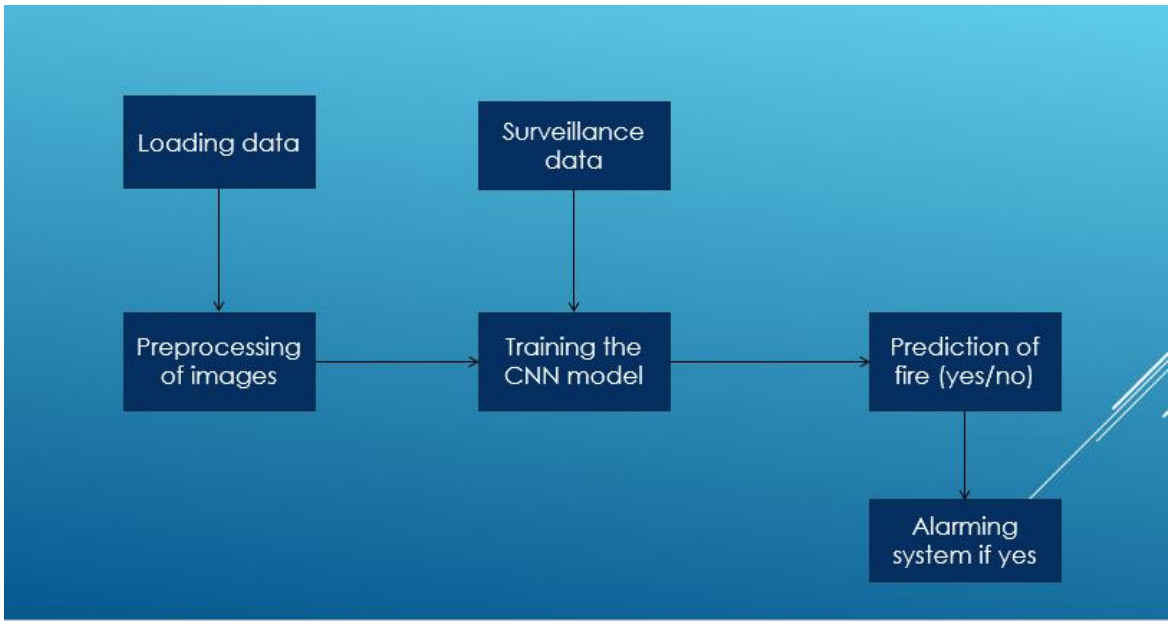
3.5.1.HARDWARE INTERFACES

The interaction between the user and the console is achieved through python capabilities/Jupyter Notebo

4.SYSTEM DESIGN

4.1 DATA FLOW DIAGRAM

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way. As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred Process Model. A DFD demonstrates business or technical process with the support of the outside data saved. plus, the data flowing from the process to another and the end results.



4.1.System Architecture

4.2. UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.2.1. CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematics of the application and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes that contain three parts:

- The upper part holds the name of the class.
- The middle part contains the attributes of the class.
- The bottom part gives the methods or operations the class can take or undertake.

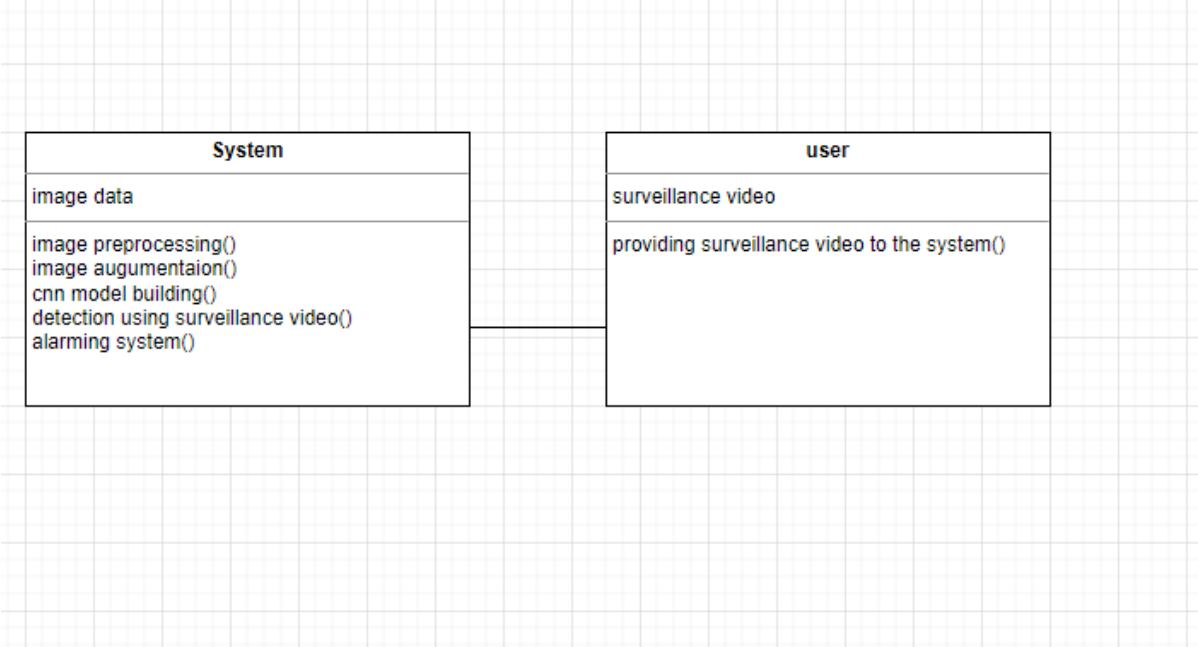


Figure 4. 2 CLASS DIAGRAM

4.2.2.USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

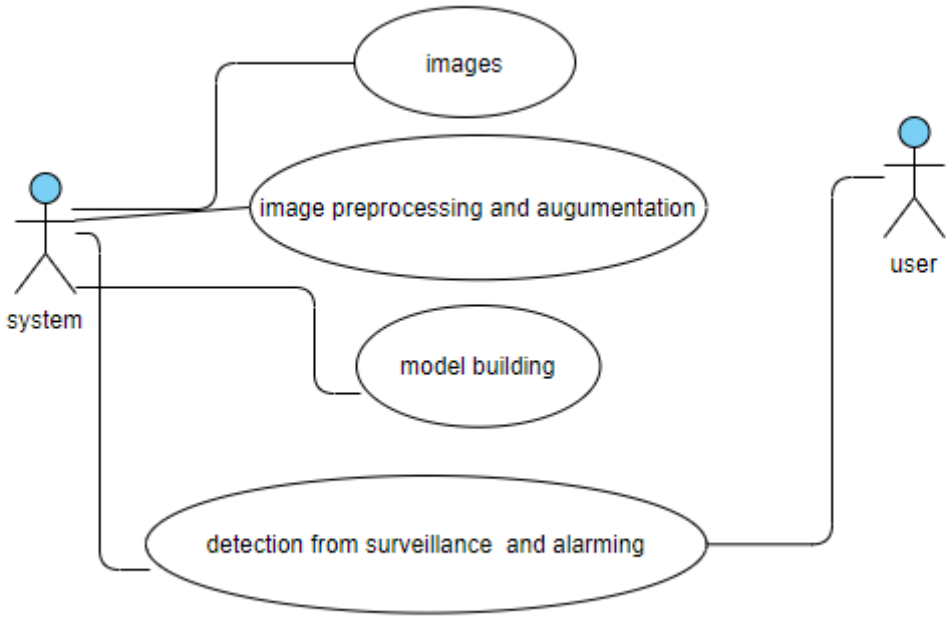


Figure 4. 3 USE CASE DIAGRAM

4.2.2. SEQUENCE DIAGRAM

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

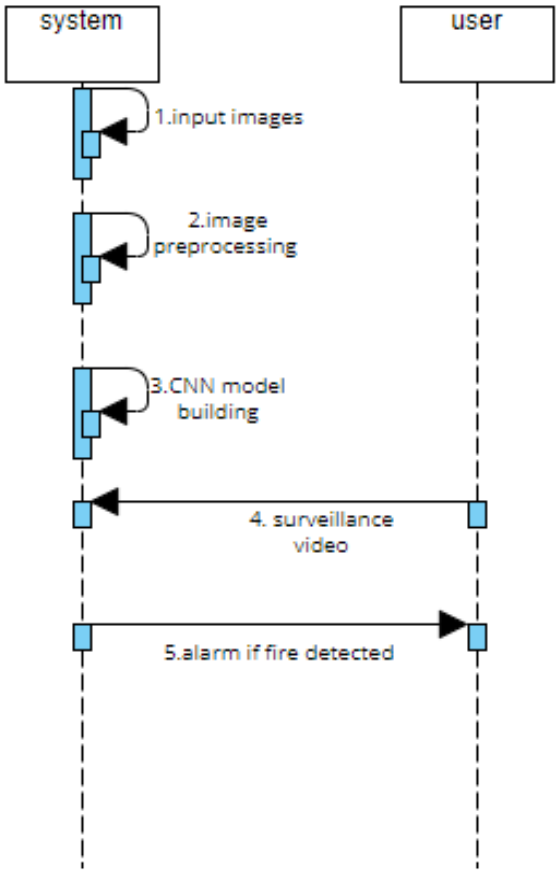


Figure 4. 4 SEQUENCE DIAGRAM

4.2.3. ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

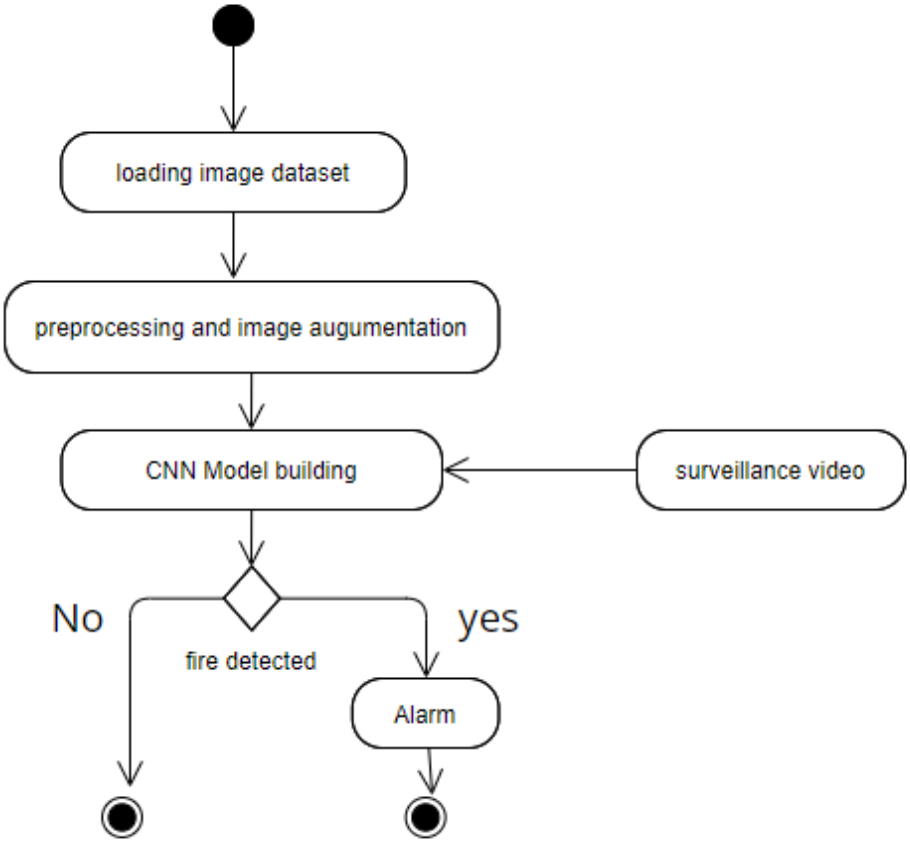


Figure 4. 5 ACTIVITY DIAGRAM

5.IMPLEMENTATION

5.1.PYTHON

Python is a general-purpose language. It has wide range of applications from Web development (like Django and Bottle), scientific and mathematical computing (Orange, Sympy, NumPy) to desktop graphical user Interfaces (Pygame, Panda 10) The syntax of the language is clean, and length of the code is relatively short. It is fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

HISTORY OF PYTHON

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

WHY PYTHON WAS CREATED

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy- to understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

5.1 InceptionNet V3 ALGORITHM

The Inception V3 is a deep learning model based on Convolutional Neural Networks, which is used for image classification. The inception V3 is a superior version of the basic model Inception V1 which was introduced as GoogLeNet in 2014. As the name suggests it was developed by a team at Google.

The inception V3 is just the advanced and optimized version of the inception V1 model. The Inception V3 model used several techniques for optimizing the network for better model adaptation.

- It has higher efficiency
- It has a deeper network compared to the Inception V1 and V2 models, but its speed isn't compromised.
- It is computationally less expensive.
- It uses auxiliary Classifiers as regularizes.

The inception v3 model was released in the year 2015, it has a total of 42 layers and a lower error rate than its predecessors.

The major modifications done on the Inception V3 model are

1. Factorization into Smaller Convolutions
2. Spatial Factorization into Asymmetric Convolutions
3. Utility of Auxiliary Classifiers
4. Efficient Grid Size Reduction

One of the major assets of the Inception V1 model was the generous dimension reduction. To make it even better, the larger Convolutions in the model were factorized into smaller Convolutions.

Traditionally max pooling and average pooling were used to reduce the grid size of the feature maps. In the inception V3 model, in order to reduce the grid size efficiently the activation dimension of the network filters is expanded.

Sample code

```
# Building a cnn model with adam optimizer
#initilizing cnn
#input
cnn=tf.keras.models.Sequential()

#hidden layers

#first layer
#convolutional layer
cnn.add(tf.keras.layers.Conv2D(filters=32,kernel_size=3,padding='valid',activat
ion='relu',input_shape=[256,256,3]))
#max pooling- decreasing the size of images by extraxting only the required
features from the image
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

#second layer
#convolutional layer
cnn.add(tf.keras.layers.Conv2D(filters=42,kernel_size=3,padding='valid',activat
ion='relu'))
#max pooling- decreasing the size of images by extraxting only the required
features from the image
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

#third layer
#convolutional layer
cnn.add(tf.keras.layers.Conv2D(filters=64,kernel_size=3,padding='valid',activat
ion='relu'))
#max pooling- decreasing the size of images by extraxting only the required
features from the image
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))
# flattening and fully connected layer
cnn.add(tf.keras.layers.Flatten())
cnn.add(tf.keras.layers.Dense(units=128,activation='relu'))

#output layer- activation here is either sigmoid(binary) or
softmax(categorical)
cnn.add(tf.keras.layers.Dense(units=2,activation='softmax'))
checkpoint=tf.keras.callbacks.ModelCheckpoint(r'C:\Users\perik\Desktop\major\fi
re-detection-master\model\acc_model-5.h5', mode='max',
monitor='val_acc',save_best_only=True,verbose=1,save_freq='epoch')
callbacks=[checkpoint]
# compile and train
#optimizer to reduce the errors,
cnn.compile(optimizer='Adam',loss='categorical_crossentropy',metrics=['acc'])
cnn.fit(train,epochs=20,validation_data=test,callbacks=callbacks)
```

```
#inceptionNet V3 with adam optimizer
from tensorflow.keras.applications.inception_v3 import InceptionV3
 from tensorflow.keras.preprocessing import image
 from tensorflow.keras.models import Model
 from tensorflow.keras.layers import Dense, GlobalAveragePooling2D,
 Input, Dropout



 input_tensor = Input(shape=(256, 256, 3))


 base_model = InceptionV3(input_tensor=input_tensor, weights='imagenet',
 include_top=False)


 # add a global spatial average pooling layer
 x = base_model.output
 x = GlobalAveragePooling2D()(x)
 x = Dense(2048, activation='relu')(x)
 x = Dropout(0.25)(x)
 x = Dense(1024, activation='relu')(x)
 x = Dropout(0.2)(x)
 predictions = Dense(2, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
 for layer in base_model.layers:
     layer.trainable = False
 model.compile(optimizer='adam', loss='categorical_crossentropy',
 metrics=['acc'])
 checkpoint=tf.keras.callbacks.ModelCheckpoint(
   r'C:\Users\perik\Desktop\major\fire-
   detection-master\model\acc_model_6.h5',
   mode='max',
   monitor='val_acc',save_best_only=True,verbos
   e=1,save_freq='epoch')
callbacks=[checkpoint]
history = model.fit(train,epochs =
20,validation_data = test,callbacks=[callbacks])
```

5.1.1 INCEPTION NET V3 ARCHITECTURE

Inception Net v3 is a variant of Inception Net model which has 93 Convolution layers (convo 2d)along with a mixed combination of Max Pool , Average Pool and global average pool layer. It also includes various batch normalization layers, along with dense and dropout layers.

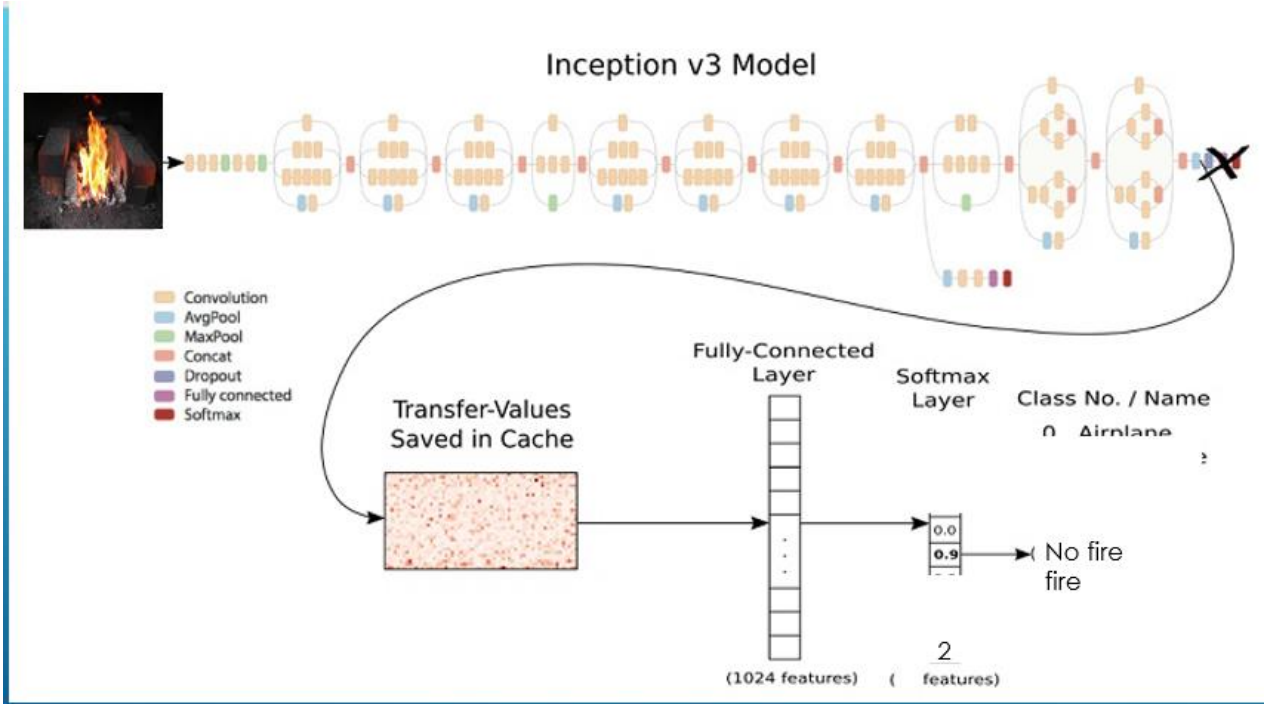


Figure 5. 1 INCEPTION NET V3 ARCHITECTURE

Convolutional layer (convo_2d):

2D convolution layer (e.g. spatial convolution over images). This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. Then these outputs of the previous layers are passed to the next layers. The input is given as a tensor with several inputs x height x width x channels of input. Then the layers perform a few mathematical operations and extract the features, using convo2D. This is repeated layer after layer which simulates the working of brain neurons.

Since it is a fully connected network all the output gets filtered and combined as a single output in the output layer.

Pooling :

Pooling in convolutional neural networks is a technique for generalizing features extracted by convolutional filters and helping the network recognize features independent of their location in the image.

Flatten: Return a copy of the array collapsed into one dimension.

Dense Layer:

It is used to classify image based on output from convolutional layers. Each Layer in the Neural Network contains neurons, which compute the weighted average of its input and this weighted average is passed through a non-linear function, called as an "activation function.

Drop Out:

Dropout is a technique where randomly selected neurons are ignored during training. They are "dropped-out" randomly. It means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.

Batch Normalization:

Normalization is a data pre-processing technique for converting numerical data to a common scale while changing the form of the data. When we feed data into a machine learning or deep learning system, we usually modify the numbers to a balanced scale. Normalization is performed in part to guarantee that our model can generalize appropriately.

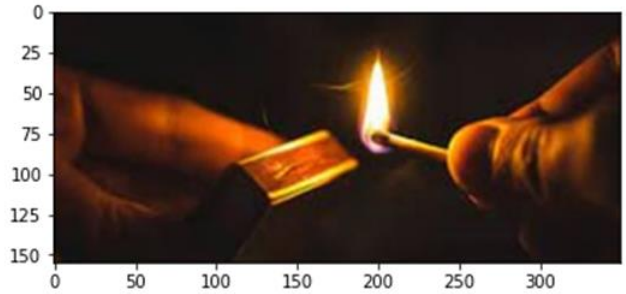
6.OUTPUT SCREENS

CNN model performance

```
#evaluating the model
#fire image
print('cnn model with adam optimizer:\n')
img1=r'C:\Users\perik\Desktop\major\fire-detection-master\Dataset 3\Train\Fire\image_910.jpg'
checking(img1,0)
```

```
cnn model with adam optimizer:

fire
```




```
img2=r'C:\Users\perik\Desktop\major\fire-detection-master\Dataset 3\Test\Neutral\image_100.jpg'
checking(img2,0)
```

```
fire
```

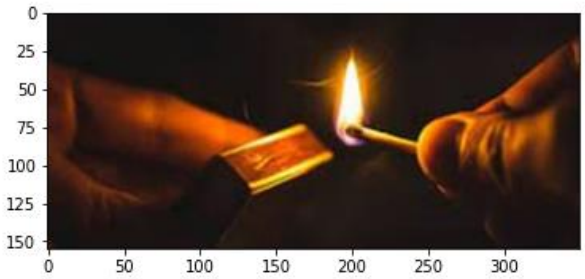
Fig 6.1. Predictions made by the CNN model built

InceptionNet V3 Performance

```
print('inceptionV3 model with adam optimizer:\n')
img1=r'C:\Users\perik\Desktop\major\fire-detection-master\Dataset 3\Train\Fire\image_910.jpg'
checking(img1,1)
```

```
inceptionV3 model with adam optimizer:

fire
```




```
img2=r'C:\Users\perik\Desktop\major\fire-detection-master\Dataset 3\Test\Neutral\image_100.jpg'
checking(img2,1)
```

```
no fire
```

Fig 6.2. Predictions made by the inception v3 model

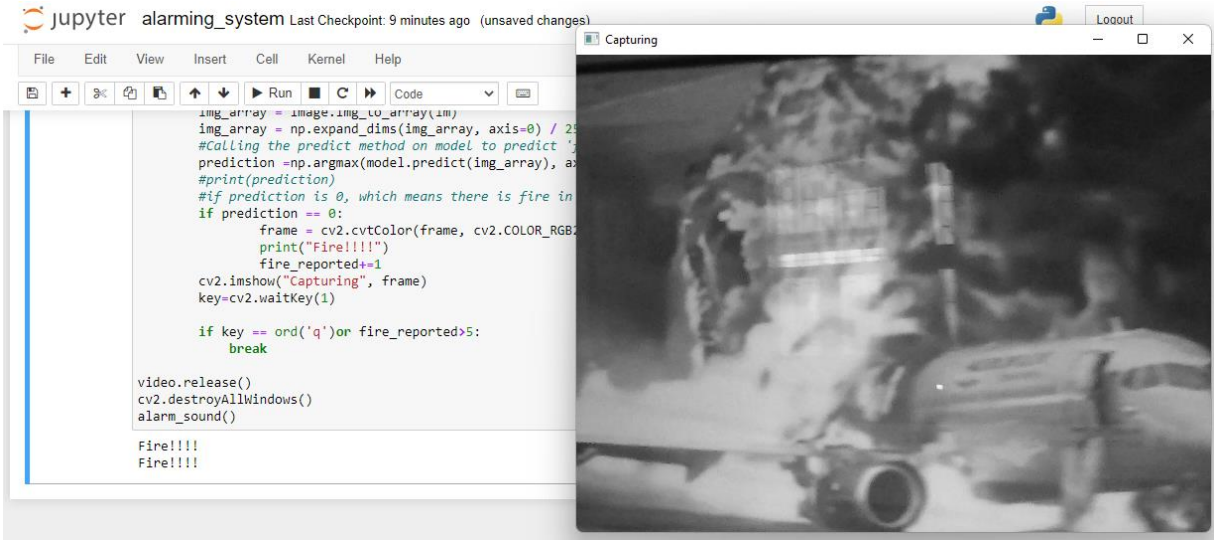


Fig 6.3. The alarming system upon detection of fire.

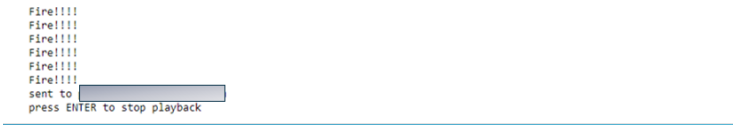


Fig 6.4. Output displayed after fire detection

Fig. 6.5. Email received at the owner's end indicating fire.

7. TESTING

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs, and improving performance.

Software testing provides an independent view and objective of the software and gives surety of the fitness of the software. It involves testing of all components under the required services to confirm whether it is satisfying the specified requirements or not. The process is also providing the client with information about the quality of the software.

Testing is mandatory because it will be a dangerous situation if the software fails any time due to lack of testing. So, without testing software cannot be deployed to the end-user.

Images used for testing:

Fig. 7.1.Image_31- forest fire

Fig 7.2. Image_6- lamp

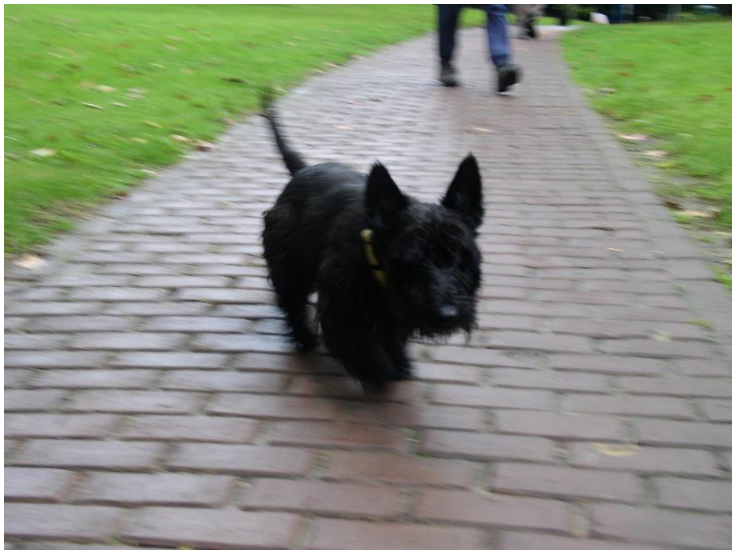


Fig 7.3 Image_2- dog

Fig 7.3. Image_100- sunset

Fig 7.4 Image_910- matchstick with fire

Table indicating the performance of the models:

7.1.Test Table

image	Cnn model	InceptionNet v3
image_910-matchstick with fire	pass	pass
image_100-sunset	fail	pass
Image_6 –lamp	fail	pass
Image_31 – forest fire	fail	pass
Image_2 –dog	pass	pass

Through this we can observe that the performance of the Inception Net V3 model is more accurate than the CNN model with Adam optimizer.

8. CONCLUSION

This proposed system provides the utilization of the Inception Net V3 model to detect fire. The project's main goal is to increase classification performance while reducing time consumption. Inception Net V3 is used to enhance the speed than more any other models like Yolo, ResNet, and many more.

Performing the real-time testing of the classification model by using the surveillance camera and detection of fire has been made with the help of OpenCV .also an alarming model has been constructed indicating the fire which is an alarm is triggered as well as an email is sent to the owner indicating the fire.

9. FUTURE ENHANCEMENT

An efficient model can be built which is more cost-efficient and takes less time to train the model, thereby reducing the training time and reducing the complexity of the model.

Images can be further processed which can make the training easier.

This project can be extended by adding sensors that can be activated when a fire is detected which can spray water around the fire and reduce the damage caused by it.

Also if we can detect the cause and type of fire, specified extinguishers can be used to reduce the fire and safeguarding ourselves.

10.REFERENCES

[1]. National Risk Survey Report - Pinkerton, FICCI (2018).

[2]. Janku P., Kominkova Oplatkova Z., Dulik T., Snopek P. and Liba J. 2018. “Fire Detection in Video Stream by Using Simple Artificial Neural”. Network. MENDEL. 24, 2 (Dec. 2018), 55–60.

[3]. Shen, D., Chen, X., Nguyen, M., & Yan, W. Q. (2018). “Flame detection using deep learning”. 2018 4th International Conference on Control, Automation and Robotics (ICCAR).

[4]. Li, C., & Bai, Y. (2018). “Fire Flame Image Detection Based on Transfer Learning”. 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS).

[5]. K. Muhammad, J. Ahmad, I. Mehmood, S. Rho and S. W. Baik, “Convolutional Neural Networks Based Fire Detection in Surveillance Videos," in IEEE Access, vol. 6, pp. 18174-18183, 2018.

[6]. S. J. Pan and Q. Yang, "A Survey on Transfer Learning," in IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010.

[7]. [Zhang, Qingjie & Xu, Jiaolong & Xu, Liang & Guo, Haifeng. (2016). “Deep Convolutional Neural Networks for Forest Fire Detection”.

[8]. K. Muhammad, J. Ahmad, Z. Lv, P. Bellavista, P. Yang and S. W. Baik, “Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications,” in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 7, pp. 1419-1434, July 2019.

[9]. Z. Jiao et al., “A Deep Learning Based Forest Fire Detection Approach Using UAV and YOLOv3,” 2019 1st International Conference on Industrial Artificial Intelligence (IAI),

Shenyang, China, 2019.

[10] .Faming Gong ,1 Chuantao Li,1 Wenjuan Gong ,1 Xin Li,1 Xiangbing Yuan,2 Yuhui Ma and Tao Song. “A RealTime Fire Detection Method from Video with Multifeature Fusion”. Hindawi, Computational Intelligence and Neuroscience, Volume 2019.

[11]. L. Shao, F. Zhu and X. Li, “Transfer Learning for Visual Categorization: A Survey,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no. 5, pp. 1019-1034, May 2015.

[12]. S. Mohd Razmi, N. Saad and V. S. Asirvadam, “Visionbased flame detection: Motion detection & fire analysis,” 2010 IEEE Student Conference on Research and Development (SCOReD), Putrajaya, 2010, pp. 187-191

11.APPENDIX (IJCRT) – Fire Detection And Alarming System

www.ijcrt.org © 2022 IJCRT | Volume 10, Issue 4 April 2022 | ISSN: 2320-2882

IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Fire Detection and Alarming System

P.Sri Sai Keerthana, T.R. Sai Shruthi, P Bhanu Prakash, M Praveen, Shashank Tiwari

Computer Science and Engineering, ACE Engineering College, Hyderabad, Telangana, India.

Abstract- History has proven that early detection of a fire and the signaling of an appropriate alarm remain significant factors in preventing large losses due to fire. Properly installed and maintained fire detection and alarm systems can help to increase the survivability of occupants and emergency responders while decreasing property losses. Considering that the majority of cities have already installed camera-monitoring systems, this encouraged us to take advantage of the availability of these systems to develop cost-effective vision detection methods. . Deep learning is an emerging concept based on artificial neural networks and has achieved exceptional results in various fields including computer vision. We plan to overcome the shortcomings of the present systems and provide an accurate and precise system to detect fires as early as possible and capable of working in various environments thereby saving innumerable lives and resources. We have made a comparative study of a general CNN (Convolutional Neural Network) model with adam optimizer and the pre-trained InceptionV3 model with adam optimizer.

Keywords – fire, deep learning, Kaggle, CNN, InceptionV3, Image Processing, Kiras, Tensorflow, Surveillance video, Alarming system.

I.INTRODUCTION

Fire accidents pose a serious threat to industries, crowded events, social gatherings, and densely populated areas that are observed across India. These kinds of incidents may cause damage to property and the environment that poses a threat to human and animal life.

According to the recent National Risk Survey Report, Fire stood at the third position overtaking corruption, terrorism, and insurgency thus posing a significant risk to our country's economy and citizens. The recent forest fires in Australia reminded the world, of the destructive capability of fire and the impending ecological disaster, by claiming millions of lives resulting in billions of dollars in damage. Early detection of fire accidents can save innumerable lives along with saving properties from permanent infrastructure damage and the consequent financial losses. To achieve high accuracy and robustness in dense urban areas, detection through local surveillance is necessary and also effective. Traditional optoelectronic fire detection systems have major disadvantages: The requirement of separate and often redundant systems, fault-prone hardware systems, regular maintenance, false alarms, and so on. Usage of sensors in hot, dusty industrial conditions is also not possible. Thus, detecting fires through surveillance video stream is one of the most feasible, cost-effective solutions suitable for the replacement of existing systems without the need for large infrastructure installation or investment.

The existing video-based machine learning models rely heavily on domain knowledge and feature engineering to achieve detection therefore, have to be updated to meet new threats. We aim to develop a classification model using Deep Learning and Transfer Learning to recognize fires in images/video frames, thus ensuring early detection and saving manual work. This model of Inception V3 can be used to detect fires in surveillance videos and send an alarming email and a siren indicating fire. Unlike existing systems, this neither requires special infrastructure for a setup like hardware-based solutions nor does it need domain knowledge and prohibitive computation for development.

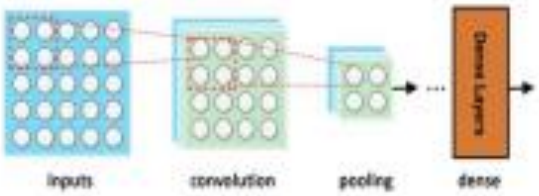


Figure 1. General Representation of CNN Model.

II. RELATED WORK

Janku P., Kominkova Oplatkova Z., Dulik T., Snopek P. and Liba J (2018) [1]. Artificial Neural Network-based approaches seen in paper [1] use the Levenberg Marquardt training algorithm for a fast solution. The accuracy of the algorithm altered between 61% to 92%. False positives ranged from 8% to 51%. This approach yielded high accuracy and a low false-positive rate, yet it requires immense domain knowledge.

Shen, D., Chen, X., Nguyen, M., & Yan, W. Q. (2018). In this paper [2] the author says that the present hardware-based detection systems offer low accuracy along with a high occurrence of false alarms consequently making it more likely to misclassify actual fires. It is also not suitable for detecting fires breaking out in large areas such as forests, warehouses, fields, buildings, or oil reservoirs. The authors used a simplified YOLO (You Only Look Once) model with 12 layers. Image augmentation techniques such as rotation, adjusting contrast, zooming in/out, saturation, and aspect ratio was used to create multiple samples of each image, forming 1720 samples in total.

Wanda Zaho (2020) [3]. Novel image fire detection algorithms based on the advanced object detection CNN models of Faster-RCNN, R-FCN, SSD, and YOLO v3 are proposed in this paper. A comparison of the proposed and current algorithms showed that the accuracy of fire detection algorithms based on object detection CNNs is higher than other algorithms. Especially, the average precision of the algorithm based on YOLO v3 reached 83.7%, which is higher than the other proposed algorithms in this paper.

Suhas G, Chetan Kumar, Abhishek B S, Digvijay Gowda K A, Prajwal R (2020). The authors in the paper [4] have experimented with various deep learning models such as Inception netV3, Inception Net V2, Res Net along with classification algorithms like decision tree, SVM, Naïve Bayes, logistic regression and have selected ResNet-50-SVM combination for implementation as it offered the best performance metric value with an accuracy of 97%.

III. SYSTEM ARCHITECTURE

The system includes image preprocessing and image augmentation on the image dataset followed by the building of the CNN models.

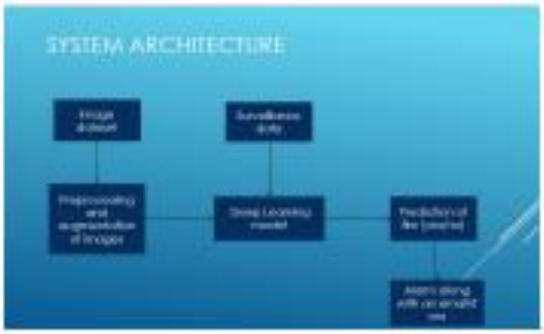


Figure 2. System Architecture

The next stage is training the model with the improved dataset obtained from the previous steps and providing the power to the model to build a classification model that can classify images into fire or no fire based on the knowledge gained by the dataset.

The result of the classification is displayed to the user, and depending on the result, further actions are taken. If the result is a fire then an alarming system is activated, and an email is sent to the concerned stakeholders.

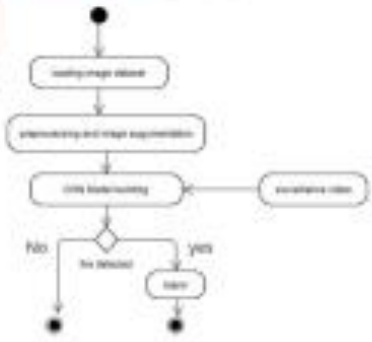


Figure 3. Activity Diagram

IV. ALGORITHMS

The algorithms included in our project are a sequential CNN model and a pre-trained transfer learning model called Inception V3 and the comparison between them is based on the optimizer Adam.

www.ijcrt.org © 2022 IJCRT | Volume 10, Issue 4 April 2022 | ISSN: 2320-2882

Convolutional Neural Network (CNN)

Convolutional neural networks are a special kind of artificial neural network that can mimic human brain activity to analyze data with supervised learning. CNN is a modified multilayer perceptron, which means a fully connected network. It consists of several layers namely, the input layer, output layer, and many hidden layers to make it happen. These hidden layers are convolutional hence the name convolutional neural networks. It offers beyond the limit abilities toper form object detection. These convolutional layers use several mathematical models to critically evaluate and analyze data. Then these outputs of the previous layers are passed to the next layers. There is a chance of overfitting since the network is fully connected. To avoid this situation, CNN exploits the hierarchical pattern in the data and sorts them according to their complexity from simpler to complex patterns engraved in the layers. The input is given as a tensor with several inputs x height x width x channels of input. Now the image is in an abstract form, then the layers convert this abstract image into a feature map. This is repeated layer after layer which simulates the working of brain neurons. Since it is a fully connected network all the output gets filtered and combined as a single output in the output layer. The number of filters is directly proportional to the feature map size. Along with the convolutional layer, it also includes a pooling layer to reduce the size of the output from the convolutional layers.

Inception V3

Inception v3 is a convolutional neural network for assisting in image analysis and object detection. It is the third edition of Google's Inception Convolutional Neural Network, originally introduced during the ImageNet Recognition Challenge. The design of Inception3 was intended to allow deeper networks while also keeping the number of parameters from growing too large. The Inception V3 model used several techniques for optimizing the network for better model adaptation. It has higher efficiency, It has a deeper network compared to the Inception V1 and V2 models, but its speed isn't compromised. It is computationally less expensive. It uses auxiliary Classifiers as regularizes. This model also involves several layers which include convolutional layers and pooling layers and it also includes batch normalization. Using batch normalization makes the network more faster and stable during the training of the network. This may require the use of larger than the normal learning rates, this, in turn, may speed up the learning process.

Adam Optimizer

Adam optimizer involves a combination of two gradient descent methodologies: Adaptive Gradient and RMS propagation. Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems). Root Mean Square Propagation (RMSProp) also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy). Adam realizes the benefits of both AdaGrad and RMSProp.

V. METHODOLOGY

The model is divided into two parts in which the first part involves data collection, data preprocessing and data augmentation and the second part involves model building, model training, and prediction using real-time data using surveillance video, and an alarming system.

Dataset, Data Preprocessing, and Data Augmentation

The dataset used for our project is taken from Kaggle named after forest fires which involves two folders namely test and train which has fire images under the folder fire and neutral images or no fire images under neutral. The total images for training were 910 fire images and 902 neutral images here we also added a few images from google images for the better training of the model. The testing dataset was 100 images each for fire and neutral for the validation of the model.

Figure 4. Training fire dataset

Choosing a proper dataset is one of the most important tasks as it acts as a base for the model to gain knowledge and classify the data.

Figure 5. Training neutral dataset

Data preprocessing performed here is image rescaling and resizing as images in the dataset are of different sizes and dimensions. Data augmentation performed here is vertical and horizontal flips along with a zoom range of .2. The main reason to perform data augmentation is to increase the dataset size from the given dataset. This is obtained by performing various operations on a single image like vertically flipping the image and this becomes new data for training, similarly, we can perform various operations such as vertical flip, zooming the image, blur, etc. to increase the dataset.

Model building, Training, Prediction, and Alarming system

We have built two models which are a simple sequential CNN model with an Adam optimizer and a pre-built transfer learning model of Inception Net V3 with an Adam optimizer. The CNN model has 3 convolutional layers with 32, 42, and 64 filters in each layer and has relu as its activation function with padding of valid along with max-pooling of size 2 at each level. The output layer with an activation function of softmax. Using the adam optimizer we have trained the model with the dataset obtained from the first step. The checkpoint was created so that we can save the best model with high accuracy during the training of the model. The Inception V3 model was a pretrained model imported from Keras which has 92 convolutional layers along with several max poolings and batch normalizations to build a better model. Using the adam optimizer we have trained this model with the same dataset. Similar to the above model the checkpoint of highest accuracy was created so that we can save the best model for classification. We initially tested with two images from ur test dataset and then predicted by using the surveillance camera for the real-time prediction using the model built using OpenCV. If the fire was detected in the video frame for more than 5 frames it trigged the alarm indicating fire and sent an email to the owner indicating the fire.

VI. RESULTS

The CNN model built gave an accuracy of 90% and the pre-trained transfer learning model of the Inception Net V3 model gave an accuracy of 98%. While we tried to predict the models with the images from the test dataset the sequential CNN model could not predict as accurately as the Inception net V3 model.

Figure 6. CNN model indicating fire for a neutral image

Figure 7. InceptionV3 model predicting a neutral image

We can observe that the Inception model can classify better than the sequential CNN model.

VII. CONCLUSIONS

The present decade is marked by huge strides in areas of processing, computation, and algorithms. This has enabled great progress in many fields including the processing of surveillance video streams for recognizing abnormal or unusual events and actions. Fire accidents have caused death and destruction all over the world, consuming countless lives and causing billions in damage. This implies that developing an accurate, early, the affordable fire-detection system is imperative Therefore, we have proposed a fire detection model for videos/video frames using transfer learning for deep learning. Based on our observation we can make use of the Inception Net V3 model along with adam optimizer for classification. Coming to the project, on the whole, it works in real-time and can send alert emails indicating fire along with a siren. It's cost-effective, reliable, robust, and accurate compared to existing optoelectronic hardware and software-based systems in the market.

ACKNOWLEDGMENT

The paper has been composed with kind assistance, guidance from Mr. Shashank Tiwari, Mrs. S.Kavitha, Dr.M.V.Vijay Saradhi, and the support of my department who have helped me in this work. We would like to thank all the people whose encouragement and support have made the fulfillment of this work conceivable.

REFERENCES

[1] Janku P., Kominkova Oplatkova Z., Dulik T., Snopek P. and Liba J. 2018. "Fire Detection in Video Stream by Using Simple Artificial Neural". Network. MENDEL. 24, 2 (Dec. 2018), 55–60.

[2] Shen, D., Chen, X., Nguyen, M., & Yan, W. Q. (2018). "Flame detection using deep learning". 2018 4th International Conference on Control, Automation and Robotics (ICCAR).

[3] Wanda Zalho (2020) "Image fire detection algorithms based on convolutional neural networks" case study in Thermal Engineering book, volume 19.

[4] Subas G, Chetan Kumar, Abhishek B S, Digvijay

Gowda K A, Prajwal R (2020)." Fire Detection Using Deep Learning". International Journal of Progressive Research in Science and Engineering Volume-1, Issue-5, August-2020.