Project Stage II Report

On

Forest Fire Detection using CNN-RF and CNN-XGBOOST Machine Learning Algorithms

submitted in partial fulfilment of the requirement for the award of degree

Bachelor of Technology

In

Computer Science and Engineering

By

AKKAMWAR AMULYA	**(20J21A0501)**
ANNAMGARI SAI KIRAN	**(20J21A0503)**
KAMUTALA CHANDRA SIDDARTHA	**(20J21A0531)**
KOMMINENI SAI MANU	**(20J21A0537)**

Under the Supervision

Of

Dr.T. PRABAKARAN, B.Tech,M.Tech Ph.D,
Head of the Department

Department of Computer Science and Engineering

JOGINPALLY B.R. ENGINEERING COLLEGE

Accredited by NAAC with A+ Grade, Recognized under Sec. 2(f) of UGC Act. 1956
Approved by AICTE and Affiliated to Jawaharlal Nehru Technological University, Hyderabad
Bhaskar Nagar, Yenkapally, Moinabad (Mandal)
R.R (Dist)-500075. T.S., India

JOGINPALLY B.R. ENGINEERING COLLEGE

Accredited by NAAC with A+ Grade, Recognized under Sec. 2(f) of UGC Act. 1956
Approved by AICTE and Affiliated to Jawaharlal Nehru Technological University, Hyderabad
Bhaskar Nagar, Yenkapally, Moinabad(Mandal)
R.R (Dist)-500075. T.S., India

JBREC
Joginpally BR Engineering College

CERTIFICATE

The project entitled “**Forest Fire Detection using CNN-RF and CNN-XGBOOST Machine Learning Algorithms**” that is been submitted by **AKKAMWAR AMULYA-20J21A0501, ANNAMGARI SAI KIRAN - 20J21A0503, KAMUTALA CHANDRA SIDDARTHA - 20J21A0531, KOMMINENI SAI MANU - 20J21A0537**in partial fulfilment of the award of Bachelor of Technology in Computer Science and Engineering to Jawaharlal Nehru Technological University Hyderabad. It is record of bonafide work carried out under our guidance and supervision. The results embodied in this project have not been submitted to any other university or Institute for award of any degree. In my opinion, this report is of standard required for the degree of Bachelor of Technology.

INTERNAL SUPERVISOR:	**HEAD OF THE DEPARTMENT**
Dr.T. PRABAKARAN **B.Tech,M.Tech Ph.D**	**Dr.T. PRABAKARAN** **B.Tech,M.Tech Ph.D**
Head of the Department	

EXTERNAL EXAMINER

DECLARATION OF THE STUDENT

I hereby declare that the project work entitled **“Forest Fire Detection using CNN-RF and CNN-XGBOOST Machine Learning Algorithms”**, developed under the supervision of **Dr.T. Prabrakaran, Designation,** JBREC and submitted to Joginpally B.R Engineering College is original and has not been submitted in part or whole for Bachelor degree to any other university.

AKKAMWAR AMULYA (20J21A0501)

ANNAMGARI SAI KIRAN (20J21A0503)

KAMUTALA CHANDRA SIDDARTHA (20J21A0531)

KOMMINENI SAI MANU (20J21A0537)

ACKNOWLEDGEMENT

We would like to take this opportunity to place it on record that this Project Report would never have taken shape but for the cooperation extended to us by certain individuals. Though it is not possible to name all of them, it would be unpardonable on our part if we do not mention some of the very important persons.

Sincerely, we acknowledge our deep sense of gratitude to our project supervisor **Dr.T. PRABRAKARAN** for his constant encouragement, help and valuable suggestions.

We express our gratitude to **Dr.T. PRABAKARAN,** HOD of COMPUTER SCIENCE and ENGINEERING for his valuable suggestions and advices.

We express our gratitude to **DR. DR.B.VENKATA RAMANA REDDY,** Principal of JOGINPALLY B.R ENGINEERING COLLEGE for his valuable suggestions and advices. We also extend our thanks to other Faculty members for their cooperation during our Project Report.

Finally we would like to thank our parents and friends for their cooperation to complete this Project Report.

AKKAMWAR AMULYA (20J21A0501)

ANNAMGARI SAI KIRAN (20J21A0503)

KAMUTALA CHANDRA SIDDARTHA (20J21A0531)

KOMMINENI SAI MANU (20J21A0537)

ABSTRACT

Early forest fire detection is of great importance to avoid the huge damage of forests caused by fires. Early fire detection focuses on smoke detection. The forest area is gradually decreased because of increasing forest fire and human activities. The satellite sensor is used to collect the forest thermal image in different places and analyze the data in these images to detect the fire region if they occur. Image processing technique can effectively predict the fire in the forest. The input image is pre-processed to enhance the image quality, because the input image has the noise, so the pre-processing technique is used to eliminate the noise in this system and enhance the image quality. The pre-processed image is taking to the segmentation process; it processes the image to adjacent the forest sub-area. In this system, the affected area is separately detected, and it gives the accurate forest fire in this system because the output image intensity is better to stabilize the average value of the image. In our proposed system we propose a deep learning method that uses a Convolutional Neural Network (CNN) to predict the forest fire detection. The convolutional layer is the main building block of the convolutional neural network. Usually, the layers of the network are fully connected in which a neuron in the next layer is connected to all the neurons in the previous layer. We are going to detect the fire in the forest result based on the accuracy which we get in train and test of the dataset based CNN algorithm using that we show the graph result.

LIST OF CONTENTS

1. INTRODUCTION

Early detection of wildfires is critical to the safety and security of environmental spaces and is one of the important and most large challenges in the government sector and forest fire managers. Forest fire is the important one is decreasing the space of the forest area. This fire detection technique also reduces human protocols and helps to monitor and protect the areas that are hard to protect. The new technique is used to facilitate the implementation of systems that allow monitoring is efficient in detailed areas, regardless of the state of the atmosphere or daytime. The satellite sensor is used to capture the forest fire image, but it has an increasing range of time resolution and space of the forest area. Satellite images also gave a fire-monitoring tool, management, and finding the damaged tool for compliance with burn areas to understand a favorable fire range. The principle of classifying this fire, such as materials from the original fire, is to check the color consistency. The proposed algorithm has rectified this problem and reduce the error. It not only detects fires but also distinguishes fires such as fire and materials. The parameters that were adopted in our proposed system operation to analyze the forest fire, threshold value, the detection of matrix value, and the differential matrix value of the system. The forests as a whole have been greatly endangered by human activities. Excessive growth in population and rapid urbanization have led to the encroachment of forest areas for building homes, factories, bridges etc. Besides, lot of illegal activities such as poaching, cutting of trees etc. take place in forest areas. Almost all Maoist activities root from the unprotected forest areas. Forest rangers can control only a limited area due to inaccessibility, lack of necessary equipment and shortage of manpower. Also, the constant monitoring of these forests is very tedious and nearly impossible. This module senses human and animal activity using infrared thermal imaging cameras. The IR sensor is used to monitor the entry and exit of humans or animals. The approximate position of animals in the forest is measure using the ultrasonic sensor. Furthermore, temperature and humidity readings are also collected using the DHT-11 sensor. The entire system is charged using a solar cell. Therefore, this module also detects forest fires at the initial level and helps determine areas of the forest that need more attention. The systems are aimed at aiding the forest rangers so that they can adequately and efficiently use their resources to effectively monitor the forest. Forest fires can potentially result in a great number of environmental disasters, causing vast economical and ecological losses as well as endangering human lives. In order to preserve natural resources and protect human safety and properties, forest fire monitoring and detection have become a significant solution, which attract an increasing interest around the world. Especially, the growth number of large scale worldwide forest fires has made automatic fire detection as an important technique for the early fire alarm. Conventional forest fire detection techniques make use of watchtowers and human observers to search and observe fires in hazardous environments. It consumes tremendous labour forces, threatens observers' safety and costs a great deal of time. Owing to the development of modern technologies, more advanced forest fire detection approaches integrating remote sensing techniques with various platforms (such as satellites, ground-based equipments, and aircrafts) are designed to overcome drawbacks of traditional methods. Particularly, due to their rapid maneuverability and improved personnel safety, there is an increasing demand to make unmanned aerial vehicles (UAVs) serve as powerful tools for operational fire-fighting. Recent decades, growing efforts have been devoted to the application of UAVs for forest fire monitoring and detection A typical UAV-based forest fire surveillance system is illustrated. which is composed of a team of UAVs, different kinds of onboard sensors, and a central ground station. The goals are to take advantages of UAVs to detect and track fires, predict their propagation, and supply realtime fire information to human firefighters and even to execute fire extinguishment with UAVs. The system can fulfil the missions of fire monitoring (search a potential fire), detection (find potential fire and produce fire alarm to firefighting staff), diagnosis (compute parameters of the fire position, extent and evolution), and prognosis (predict the fire propagation). Forest fire monitoring is to find the possible occurrence of fire before it has appeared, while fire detection is to confirm whether there is a real fire in progress. Fire diagnosis is for the purpose of finding detailed data of fire. Fire prognosis aims to track and predict the fire propagation based on real time information of weather, vegetation composition of forest and fire parameters. In order to complete the above-mentioned tasks with minimum interference of human operators, the specific activities are the development of UAV frames (fixed wing and rotary-wing types) carrying the necessary payload (remote sensing sensors for day-time, night-time, and all weather conditions) for fire detection and surveillance; Remote sensing technologies for fires monitoring and detection; Sensors fusion and image processing techniques for rapid fire detection, decision-making, and localization;

Guidance, navigation, control (GNC) algorithms for single UAV and multiple UAVs for monitoring, detection, tracking and prediction of fire development, and fire extinguishing operations; Cooperative localization, deployment, and control strategies of UAVs for optimal coverage of fire areas for precise and rapid fire tracking, prediction, and assistance/guidance of fire fighting; Autonomous and reliable path planning and re-planning strategies before and after fire being detected based on the fire development situations; Ground station which includes satellite and wireless communications, ground computation, image processing, visualization for fire detection, tracking and prediction with automatic fire alarm and for safe and efficient operation of UAVs systems during the entire mission. : Forest fires are very dangerous. Once they become disasters, it is very difficult to extinguish. In this paper, an unmanned aerial vehicle (UAV) image-based forest fire detection approach is proposed. Firstly, the local binary pattern (LBP) feature extraction and support vector machine (SVM) classifier are used for smoke detection, so as to make a preliminary discrimination of forest fire. In order to accurately identify it in the early stage of the fire, according to the convolutional neural network (CNN), it has the characteristics of reducing the number of parameters and improving the training performance through local receptive domain, weight sharing and pooling. This paper proposes another method for detecting forest fires in convolutional neural networks. Image preprocessing operations such as histogram equalization and smooth low-pass filtering are performed prior to inserting the image into the CNN network. The effectiveness of the proposed method is verified by detecting real forest fire images. UAVs are a new type of aviation platform. In recent years, with the continuous maturity of technology, it has been applied to many fields such as meteorological detection, disaster monitoring, power line inspection, and post-disaster rescue. In particular, the lightweight and small size UAVs have the characteristics of low cost, simple operation and flexible maneuvering. They can adjust the work plan and onboard remote sensing equipment according to the real-time situations on site, which is very suitable for the detection of forest fires. At present, relevant theoretical research on forest fire detection based on computational vision has been carried out worldwide. Compared with thermal imaging systems such as hyper spectral sensors, infrared sensors, and multi-spectral sensors, UAVs equipped with ordinary cameras for forest fire identification research have the advantages of low price, common application, and simple operation and the like.

Forest fire detection mainly includes flame detection and smoke detection. There are many different methods of target detection, and we must find a suitable way to detect forest fires. First of all, the smoke characteristics of forest fires are easier to observe than flames. That is, more significant. Therefore, research on smoke detection is first considered. After comparison, the LBP is selected to extract the smoke texture features. Finally, the SVM classifier is used to detect the forest smoke. In addition, early forest fire detection make a little more sense, so we consider simultaneous detection smoke and flame. Due to its multi-level network structure and thousands of network nodes, the neural network can implement the approximation of complex functions and represent the distribution characteristics of the input data. Among them, the convolutional neural network reduces the number of parameters by core ideas such as local receptive domain weight sharing and pooling to improve training performance. The characteristics of CNN make it have many advantages in speech recognition and image processing, especially for multi-dimensional input vector images that can be directly input into the model, avoiding the complexity of data reconstruction in feature extraction and classification. Due to the fact that fire is one of the most harmful natural hazards affecting everyday life around the world, early fire warning systems have attracted particular attention recently. The most advanced approaches in automatic early forest fire detection are based on space borne (satellite), airborne (UAVs - Unmanned Aerial Vehicles) or terrestrialbased systems. Among these, terrestrial systems based on CCD video cameras are considered as the most promising technology for automatic fire detection due to their low cost, high resolution, short time response and easy confirmation of the alarm by a human operator through the surveillance monitor. For this reason, video-based flame detection techniques have been widely investigated during the last decade. The main challenge in video-based flame detection lies in the modeling of the chaotic and complex nature of the fire phenomenon and the large variations of flame appearance in video. To address this problem many researchers use the motion characteristics of flame as well as the spatial distribution of fire colors in the scene or they try to combine both temporal and spatial characteristics. However, many natural objects have similar behavior with fire, e.g., the sun, various artificial lights or light reflections on various surfaces, dust particles etc, which can often be mistakenly detected as flames. Moreover, scene complexity and low video quality also affect the robustness of vision-based flame detection algorithms, thus, increasing the false alarm rate. On the other hand, dynamic texture analysis has been successfully applied in the past for the classification of video sequences in multimedia databases. A dynamic texture in video can be simply defined as a texture with motion, i.e., a spatially and time-varying visual pattern that forms an image sequence or part of an image sequence with a certain temporal stationary. While dynamic texture analysis is also applied to the categorization of sequences containing flame, smoke, steam etc, these general techniques are not used in practical fire detection algorithms due to their high computational cost. Furthermore, most of the existing dynamic texture categorization methods are used to model a complete video sequence or a manually selected image region in a video; hence, they cannot provide to a fire detection system neither any information regarding the exact localization of the fire in the scene nor the time of the fire incident. Every year a large number of wildfires all over the world burn forested lands causing adverse ecological, economic and social impacts. Beyond taking precautionary measures, early warning and immediate response are the only ways to avoid great losses. To this end, in this paper we propose a computer vision approach for fire-flame detection to be used by an early-warning fire monitoring system. Initially, candidate fire regions in a frame are defined using background subtraction and color analysis based on a non-parametric model. Subsequently, the fire behavior is modelled by employing various spatio-temporal features such as color probability, flickering, spatial and spatiotemporal energy, while dynamic texture analysis is applied in each candidate region using linear dynamical systems and a bag of systems approach. To increase the robustness of the algorithm, the spatio-temporal consistency energy of each candidate fire region is estimated by exploiting prior knowledge about the possible existence of fire in neighboring blocks from the current and previous video frames. Recently, a variety of sensors have been introduced for different applications such as setting off a fire alarm, vehicle obstacle detection, visualizing the interior of the human body for diagnosis, animal and ship monitoring, and surveillance. Of these applications, surveillance has primarily attracted the attention of researchers due to the enhanced embedded processing capabilities of cameras. Using smart surveillance systems, various abnormal events such as road accidents, fires, medical emergencies, etc., can be detected at early stages, and the appropriate authority can be autonomously informed. A fire is an abnormal event which can cause significant damage to lives and property within a very short time [8]. The main causes of such disasters include human error or a system failure which results in severe loss of human life and other damage. In Europe, fire disasters affect 10 000 km2 of vegetation zones each year; in North America and Russia, the damage is about 100 000 km2. Fire disasters killed 19 fire fighters and ruined 100 houses in Arizona, USA. Similarly, another forest fire in California ruined an area of land the size of 1042 km2, causing a loss of $127.35 million. According to an annual disaster report. In order to avoid such disasters, it is important to detect fires at early stages utilizing smart surveillance cameras. Two broad categories of approach can be identified for fire detection: 1) traditional fire alarms and 2) vision sensorassisted fire detection. Traditional fire alarm systems are based on sensors that require close proximity for activation, such as infrared and optical sensors. These sensors are not well suited to critical environments and need human involvement to confirm a fire in the case of an alarm, involving a visit to the location of the fire. Furthermore, such systems cannot usually provide information about the size, location, and burning degree of the fire. To overcome these limitations, numerous vision sensor-based methods have been explored by researchers in this field; these have the advantages of less human interference, faster response, affordable cost, and larger surveillance coverage. In addition, such systems can confirm a fire without requiring a visit to the fire's location, and can provide detailed information about the fire including its location, size, degree, etc. Despite these advantages, there are still some issues with these systems, e.g., the complexity of the scenes under observation, irregular lighting, and low-quality frames; researchers have made several efforts to address these aspects, taking into consideration both color and motion features. The approach is based on frame-to-frame differences, and hence cannot distinguish between fire and fire-colored moving regions. Marbach et al. investigated the YUV color space using motion information to classify pixels into fire and non fire components. Töreyin et al. used temporal and spatial wavelet analysis to determine fire and non fire regions. Their approach uses many heuristic thresholds, which greatly restricts its real world implementation. Han and Lee compared normal frames with their color information for tunnel fire detection; this method is suitable only for static fires, as it is based on numerous parameters. Çelik and Demirel explored the YCbCr color space and presented a pixel classification method for flames. To this end, they proposed novel rules for separating the chrominance and luminance components. However, their method is unable to detect fire from a large distance or at small scales, which are important in the early detection of fires. In addition to these color space-based techniques, Borges and Izquierdo utilized the low-level features including color, skewness, and roughness in combination with a Bayes classifier for fire recognition. Rafiee et al. Investigated a multi resolution two dimensional wavelet analysis to improve the thresholding mechanism in the RGB color space. Their method reduced the rate of false alarms by considering variations in energy as well as shape; however, false alarms can be higher in this approach for the case of rigid body movements within the frames, such as the movement of a human arm in the scene. Foggia et al. Presented a modified version of based on a YUC color model, which obtained better results than the RGB version. Another similar method based on color information and an SVM classifier is presented in. This method can process 20 frames/s; however, it cannot detect a fire from a large distance or of small size, which can occur in real-world surveillance footage. Color-based methods typically generate more false alarms due to variations in shadows and brightness, and often misclassify people wearing red clothes or red vehicles. Mueller et al. Attempted to solve this issue by analyzing changes in the shape of a fire and the movement of rigid objects. Their algorithm can distinguish between rigid moving objects and a flame, based on a feature vector extracted from the optical flow and the physical behavior of a fire. Di Lascio et al. combined color and motion information for the detection of fire in surveillance videos. Dimitropoulos et al. [26] used spatio-temporal features based on texture analysis followed by an SVM classifier to classify candidate regions of the video frames into fire and nonfire. This method is heavily dependent on the parameters used; for instance, small-sized blocks increase the rate of false alarms, while larger blocks reduce its sensitivity. Similarly, the time window is also crucial to the performance of this system; smaller values reduce the detection accuracy, while larger values increase the computational complexity. These dependencies greatly affect the feasibility of this approach for implementation in real surveillance systems. Recently, Foggia et al. Proposed a real-time fire detection algorithm based on color, shape, and motion features, combined in a multi expert system. The accuracy of this approach is higher than that of other methods; however, the number of false alarms is still high, and the accuracy of fire detection can be further improved. A survey of the existing literature shows that computationally expensive methods have better accuracy, and simpler methods compromise on accuracy and the rate of false positives. Hence, there is a need to find a better tradeoff between these metrics for several application scenarios of practical interest, for which existing computationally expensive methods do not fit well. A forest is a complex ecosystem which is home for many living things and various resources. Forest fire is the most significant uncontrollable hazard in forests. They completely destroy the wealth of forest and disturbing the balance condition of fauna and flora, bio diversity, and ecology and environment. These giant size fires can spread out and change the direction rapidly from its source. Forest fires are classified according to its size, cause of fire, speed, the effect of weather on the fire and the combustible material present. Generally they are classified by their fuels as ground fires (roots and other buried organic matters), surface fires (grass, leaf and debris), ladder fires (small trees) and aerial or crown fires (tall trees). The extremely hot crown fires are worst case in forest fires because they can spread out and change their direction rapidly. Brown and Smith (2000) classified forest fires according to the severity of the fire as understory fire (not dangerous and not change the structure of the vegetation), mixed brutality fire (most noteworthy spoil in dominant vegetation), stand replacement fire (destroy the aboveground parts of vegetation and change its structure), and non-fire regime (no or little occurrence of fire). The weather condition can directly (lightning) or indirectly (dry weather and drought) contribute to the forest fires. Approximately 20 % of the lightning strikes on our earth can cause fires in the forest. The fuel, oxygen and heat source are the three important resources for forest fires. They are called as the fire triangle. Trees, plants, roots, grasses and other flammable materials can fuel the fires. The air brings in oxygen which exaggerates the fires which lead to destruction. The sources like lightning, camp fires, burning cigarettes, and hot winds provide sufficient heat to initiate a forest fire. Some of the traditional fire extinguishing methods includes Water dousing, spraying fire retardants, removal of vegetation to form fire breakers, deliberately starting fires.

2. LITERATURE SURVEY

Title: Improving Nocturnal Fire Detection With the VIIRS Day–Night Band

Year: 2016

Author: Thomas N. Polivka, Jun Wang, Luke T. Ellison, Edward J. Hyer, and Charles M. Ichoku

Methodology

The basis method for this is that pixels with visible light emission and relatively pronounced BT4 signatures are likely to be fires (or volcanoes in some cases). Initially, FILDA and AFARP begin nearly the same by screening out invalid pixels such as clouds and bad data, although FILDA does not exclude water pixels. An added step for FILDA is also filtering out pixels with a solar zenith angle less than 100◦, fully removing all twilight areas and thus focusing on the fire detection at night only. Afterward, the overlap correction is applied, and the DNB is collocated with the M-bands

Title: Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications

Year: 2018

Author: Khan Muhammad, Student Member, IEEE, Jamil Ahmad , Student Member, IEEE, Zhihan Lv , Member, IEEE, Paolo Bellavista , Senior Member, IEEE, Po Yang , Member, IEEE, and Sung Wook Baik , Member, IEEE

Methodology

The embedded processing capabilities of smart cameras have given rise to intelligent CCTV surveillance systems. Various abnormal events such as accidents, medical emergencies, and fires can be detected using these smart cameras. Of these, fire is the most dangerous abnormal event, as failing to control it at an early stage can result in huge disasters, leading to human, ecological and economic losses. Inspired by the great potential of CNNs, we propose a lightweight CNN based on the SqueezeNet architecture for fire detection in CCTV surveillance networks. Our approach can both localize fire and identify the object under surveillance.

Title: Spatio-Temporal Flame Modeling and Dynamic Texture Analysis for Automatic Video-Based Fire Detection

Year: 2013

Author: Kosmas Dimitropoulos, Panagiotis Barmpoutis and Nikos Grammalidis

Methodology

We proposed an algorithm for real time videobased flame detection. By modeling both the behavior of the fire using various spatio-temporal features and the temporal evolution of the pixels' intensities in a candidate image block through dynamic texture analysis, we showed that we can have high detection rates, while reducing the false alarms caused by fire-colored moving objects. The use of spatiotemporal consistency energy increases the robustness of the algorithm by exploiting prior knowledge about the possible existence of fire in neighboring blocks from the current and previous video frames. Experimental results with thirty seven videos containing actual fire and moving fire colored objects showed that the proposed algorithm outperforms existing flame detection algorithms.

Title: The cartographer and the map reader. This entry focuses on the mapping of time and spatiotemporal data, the types of time, current methods of mapping, **Year**: 2013

Author: Yogesh Deshpande, Crispin Lobo, Jahnavi Patel, Krishi Savla, Prof. Shivani Bhattacharjee

Methodology

This system aims to efficiently monitor the forest. The forest rangers and other concerned authorities can effectively distribute their resources based on requirements determined by this system. The forest monitoring unit will be placed at optimum locations so as to provide useful data. This system will make monitoring the forest much more efficient and easier.

Title: UAV Image-based Forest Fire Detection Approach Using Convolutional Neural Network

Year: 2019

Author: Yanhong Chen, Youmin Zhang, Jing Xin, Guangyi Wang, Lingxia Mu, Yingmin Yi, Han Liu1, Ding Liu

Methodology

A smoke detection method combining LBP feature extraction and SVM classifier, using texture features for smoke detection; a forest fire detection method based on convolutional neural network: smoke and flame detection are performed by two CNN models. Based on the result of feature extraction, the “circular equivalent rotation-invariant LBP mode (tight RILBP)” is used in the detection of smoke. Compared with the CNN-9 model, the CNN-17 model reduces the complexity of the algorithm and improves the detection accuracy. In addition, the method proposed in this paper detects both smoke and flame. Finally, the effectiveness of the proposed method is verified by two sets of experiments.

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Forest fire detection images are based on fire images and non-fire images. It detects the fire accrued area in the forest. Forest fire detection in a particular area is tough to detect. The existing system detects the low level accuracy of performance based on fire occurred in the forest. The existing system doesn’t effectively classify and detect the fire in area of the forest.

DISADVANTAGES

- No comparison is made between the accuracies of several algorithm
- The overall classification accuracy was found to be the same irrespective of the kernel types.
- Occurrence of errors are more in single Feed Forward Neural Network with large no. of hidden neurons
- Processing building the model requires fast and efficient processors which is cost consuming

3.2 PROPOSED SYSTEM

The proposed model is introduced to overcome all the disadvantages that arise in the existing system. This system will increase the accuracy of the deep neural network results by classifying the forest fire image dataset using Deep learning algorithm. It enhances the performance of the overall classification results. In preprocessing method we are doing segmentation process to identify the fire accrued area. Detecting the forest fire from segmented images is to find the accuracy more reliable.

ADVANTAGES

- High performance.
- Segmentation process is easy to identify fire in the forest.
- Convolutional Neural Network is used to find the accuracy more reliable.

3.3 MODULES

- Data Selection and Loading
- Data Preprocessing
- Segmentation
- Splitting Dataset into Train and Test Data
- Classification
- Prediction
- Result Generation

3.3.1 DATA SELECTION AND LOADING

- The data selection is the process of selecting the data detect the fire
- In this project, the fire dataset is used for predicting forest fire.
- The dataset which contains the information about the forest fire with corresponding location.

3.3.2 DATA PREPROCESSING

- Image Data pre-processing is the process of getting rescale data from the dataset.
    - ❖ Resize image dataset
    - ❖ Getting data
- Resize image dataset: Rescale the fire image size into 200.
- **HSV** segmentation is implemented and separates color information (chroma) from intensity or lighting (luma). Because value is separated, you can construct a

histogram or thresholding rules using only saturation and hue. This in theory will work regardless of lighting changes in the value channel.

3.3.3 SPLITTING DATASET INTO TRAIN AND TEST DATA

- Data splitting is the act of partitioning available data into. Two portions, usually for cross-validators purposes.
- One portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
- Separating data into training and testing sets is an important part of evaluating data mining models.
- Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.

3.3.4 CLASSIFICATION

CNN In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. They have applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, brain-computer interfaces, and financial time series. CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to over fitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

3.3.5 PREDICTION

- It’s a process of forest fire from the dataset.
- This project will effectively detect the data from dataset by enhancing the performance of the overall prediction results.

3.3.6 RESULT GENERATION

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

- Accuracy
- Graph based on prediction

4. FEASIBILITY STUDY

The feasibility study is carried out to test whether the proposed system is worth being implemented. The proposed system will be selected if it is best enough in meeting the performance requirements.

The feasibility carried out mainly in three sections namely.

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility

Economic Feasibility

Economic analysis is the most frequently used method for evaluating effectiveness of the proposed system. More commonly known as cost benefit analysis. This procedure determines the benefits and saving that are expected from the system of the proposed system. The hardware in system department if sufficient for system development.

Technical Feasibility

This study centre around the system’s department hardware, software and to what extend it can support the proposed system department is having the required hardware and software there is no question of increasing the cost of implementing the proposed system. The criteria, the proposed system is technically feasible and the proposed system can be developed with the existing facility

Behavioural Feasibility

People are inherently resistant to change and need sufficient amount of training, which would result in lot of expenditure for the organization. The proposed system can generate reports with day-to-day information immediately at the user’s request, instead of getting a report, which doesn’t contain much detail.

5. SOFTWARE REQUIREMENT SPECIFICATION

Software Requirements

➢ O/S	: Windows 7.
➢ Language	: Python
➢ Front End	: Anaconda Navigator - Spyder

Hardware Requirements

➢ System	: Pentium IV 2.4 GHz
➢ Hard Disk	: 200 GB
➢ Mouse	: Logitech.
➢ Keyboard	: 110 keys enhanced
➢ Ram	: 4GB

6. SYSTEM DESIGN

6.1 UML DIAGRAMS

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

6.2 FLOW DIAGRAM

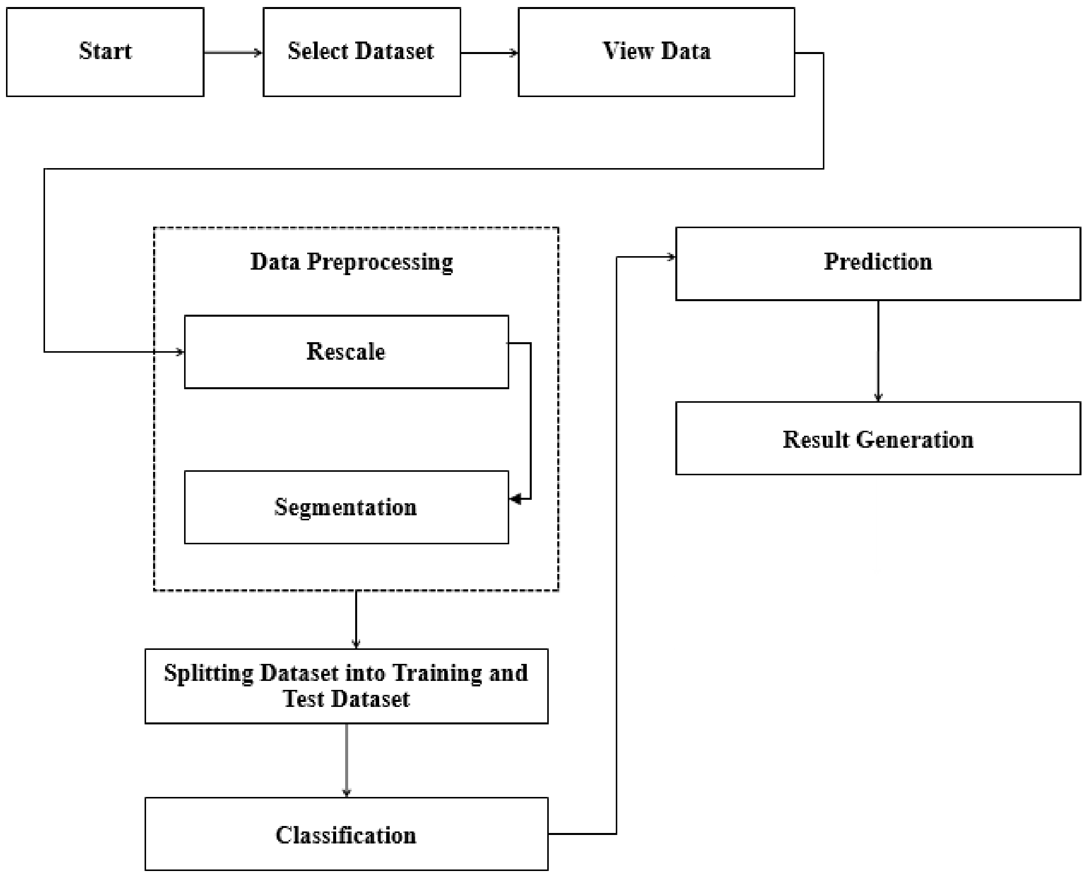


FIGURE:6.1 FLOW DIAGRAM

6.3 USE CASE DIAGRAM

The purpose of usecase diagram is to present a graphical overview of the functionality provided by a system in terms of actors , their goals (represented as use cases), and any dependencies between those use cases .The main purpose of a use case diagram is to show what system functions are performed for which actor .Roles of the actors in the system can be depicted

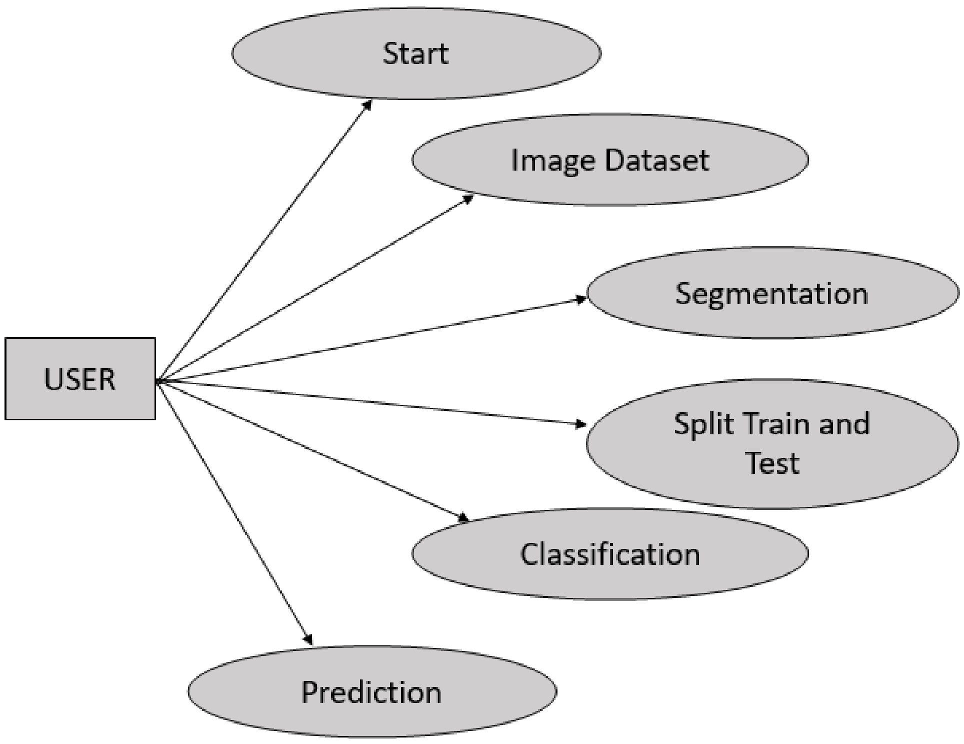


FIGURE:6.2 USE CASE DIAGRAM

ACTORS : User

USE CASES: Start, Image Dataset, Segmentation , Split Train and Test, Classification, Predication

RELATIONSHIP: Dependency

6.4 INTERACTION DIAGRAMS

6.4.1 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) 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. Sequence diagrams are sometimes called event diagrams , event scenarios, and timing diagrams

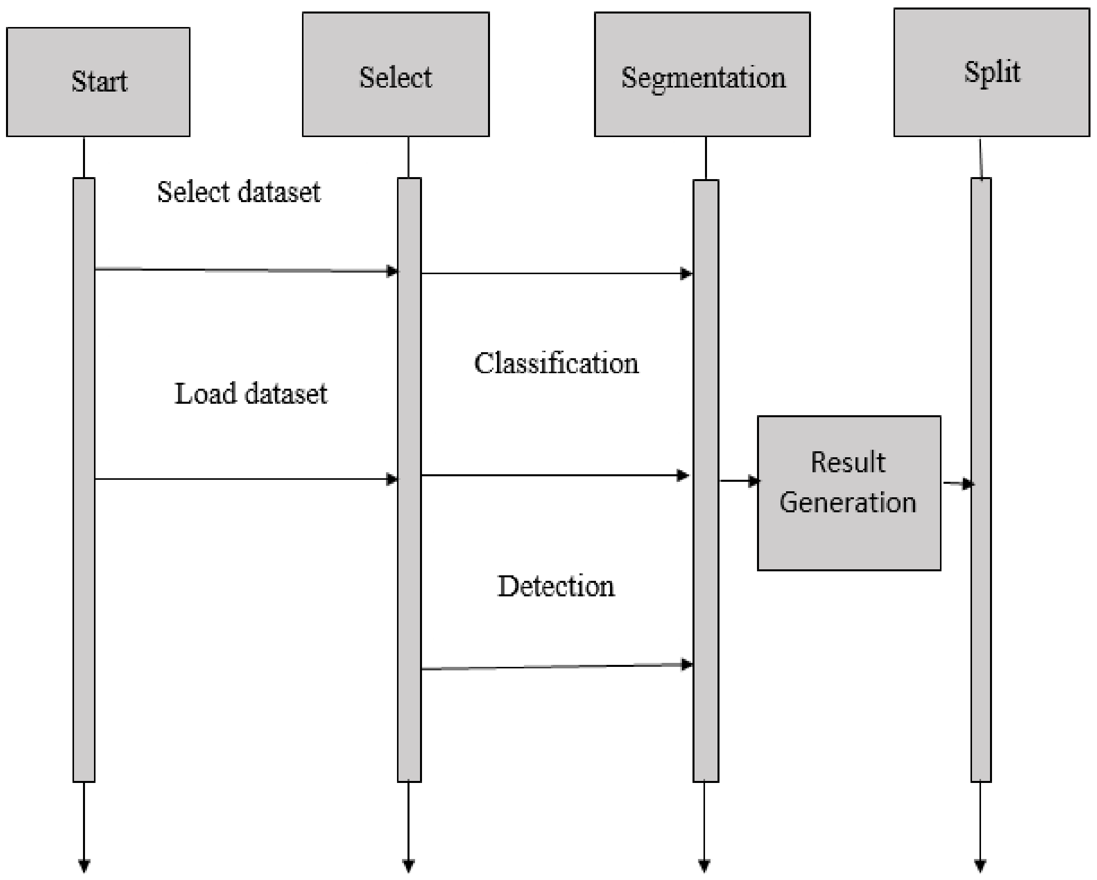


FIGURE:6.3 SEQUENCE DIAGRAM

OBJECTS : Start, Select ,Segmentation, Split, Result generation

RLATIONSHIP: Association

6.5 E-R DIAGRAM

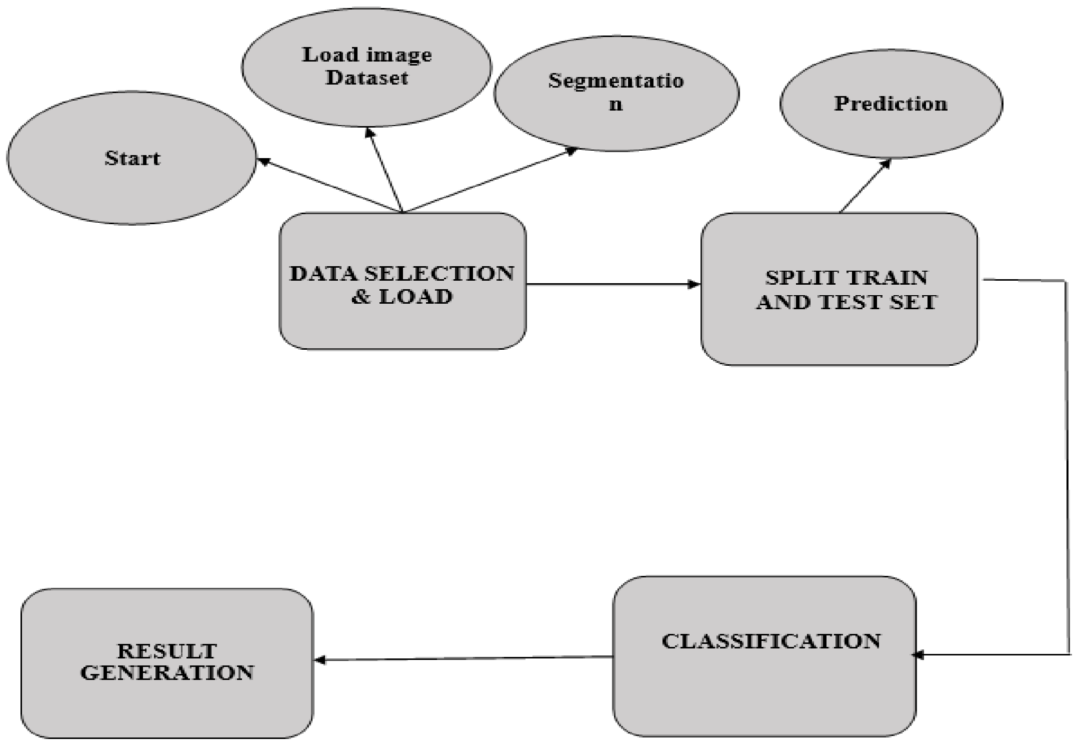


FIGURE:6.4 E-R DIAGRAM

7. SYSTEM IMPLEMENTATION

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. The people are not sure that the software is meant to make their job easier.

- ✓ The active user must be aware of the benefits of using the system
- ✓ Their confidence in the software built up
- ✓ Proper guidance is impaired to the user so that he is comfortable in using the application

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

User Training

To achieve the objectives and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Education is complementary to training. It brings life to formal training by explaining the background to the resources for them. Education involves creating the right atmosphere and motivating user staff. Education information can make training more interesting and more understandable.

Training on the Application Software

After providing the necessary basic training on the computer awareness, the users will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design, type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the data entered. This training may be different across different user groups and across different levels of hierarchy.

Operational Documentation

Once the implementation plan is decided, it is essential that the user of the system is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. Useful tips and guidance is given inside the application itself to the user. The system is developed user friendly so that the user can work the system from the tips given in the application itself.

System Maintenance

The maintenance phase of the software cycle is the time in which software performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is to make adaptable to the changes in the system environment. There may be social, technical and other environmental changes, which affect a system which is being implemented. Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance characteristics of the system. So only thru proper system maintenance procedures, the system can be adapted to cope up with these changes. Software maintenance is of course, far more than "finding mistakes".

Corrective Maintenance

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called Corrective Maintenance.

Adaptive Maintenance

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore Adaptive maintenance termed as an activity that modifies software to properly interfere with a changing environment is both necessary and commonplace.

Perceptive Maintenance

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new capabilities, modifications to existing functions, and general enhancement are received from users. To satisfy requests in this category, Perceptive maintenance is performed. This activity accounts for the majority of all efforts expended on software maintenance.

Preventive Maintenance

The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability, or to provide a better basis for future enhancements. Often called preventive maintenance, this activity is characterized by reverse engineering and re-engineering techniques.

7.1 SYSTEM ARCHITECTURE

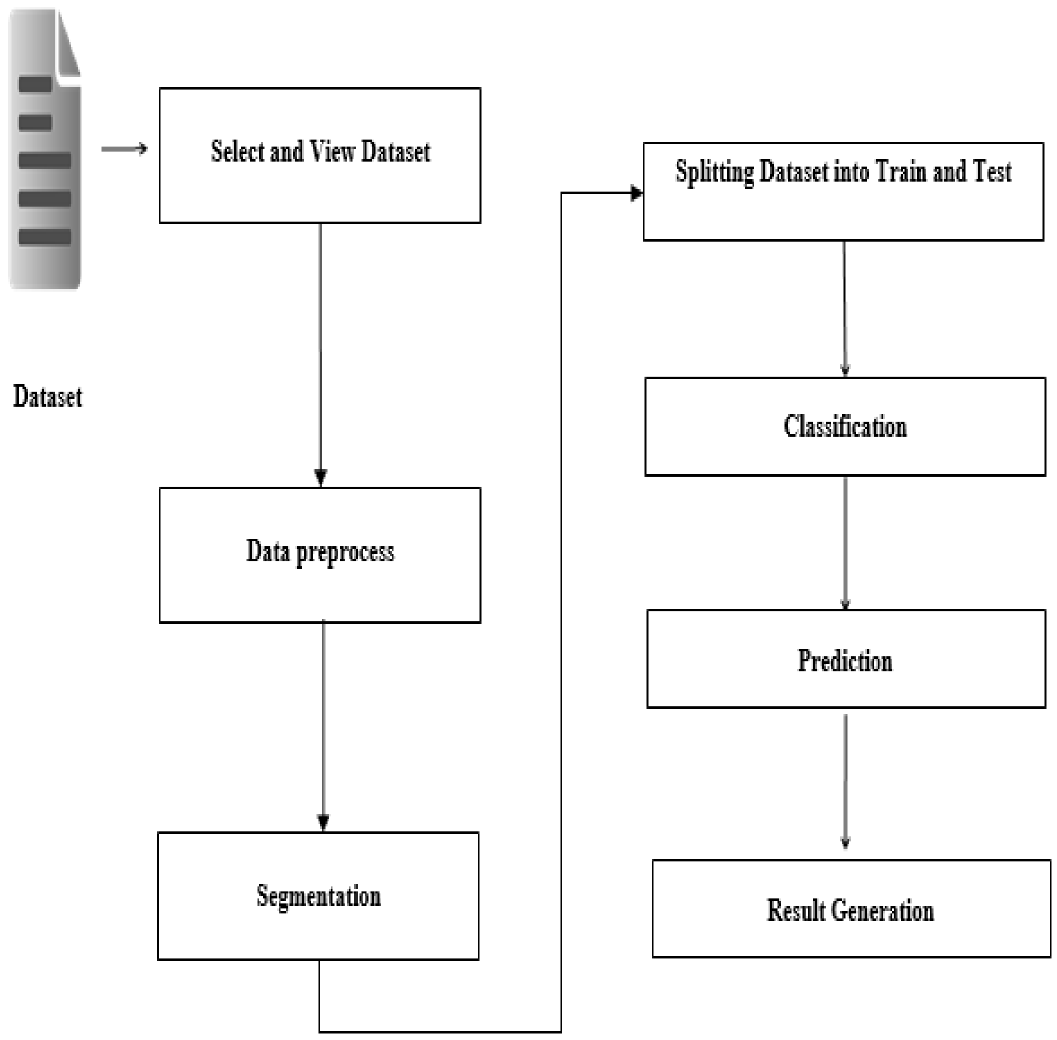


FIGURE:7.1 ARCHITECTURE DIAGRAM

Forest fire detection using CNN-RF (Random Forest) and CNN-XGBoost (Extreme Gradient Boosting) involves a sophisticated system architecture that integrates deep learning techniques with ensemble learning methods to accurately identify and predict the occurrence of forest fires. This architecture typically consists of several interconnected components, each serving a specific purpose in the detection process.

The architecture begins with data acquisition, where various sources such as satellite imagery, aerial surveys, or ground-based sensors capture images of forested areas. These images serve as input data for the detection system. Preprocessing techniques may be applied to enhance the quality and usability of the images, including normalization, resizing, and noise reduction, ensuring consistency and compatibility across different data sources.

Once preprocessed, the images are fed into the Convolutional Neural Network (CNN) component of the architecture. The CNN is responsible for feature extraction and representation learning from the input images. It consists of multiple layers of convolutional, pooling, and activation functions, which automatically learn relevant features such as textures, shapes, and patterns indicative of fire presence. The CNN architecture is designed to be deep and complex, allowing it to capture intricate relationships within the image data.

Following feature extraction, the output from the CNN undergoes further processing through either the Random Forest (RF) or Extreme Gradient Boosting (XGBoost) ensemble learning models. These models are trained using a combination of CNN-extracted features and labeled data indicating the presence or absence of fires in the training images. RF and XGBoost are popular ensemble learning techniques known for their robustness and ability to handle complex datasets.

In the case of CNN-RF architecture, the output features from the CNN are used as input features for the RF classifier. The RF model aggregates predictions from multiple decision trees, each trained on a subset of the input data, to make a final prediction regarding fire occurrence. This ensemble approach helps reduce overfitting and improves the overall accuracy of the detection system.

Similarly, in the CNN-XGBoost architecture, the output features from the CNN are utilized as input features for the XGBoost classifier. XGBoost employs gradient boosting, sequentially adding decision trees to minimize the loss function, thereby optimizing the model's predictive performance. By iteratively refining the predictions, XGBoost enhances the model's ability to discern subtle patterns and variations in the input data.

The final stage of the architecture involves post-processing and decision-making based on the output of the ensemble classifiers. Thresholding techniques may be applied to convert probability scores into binary predictions (fire/no fire). Additionally, spatial and temporal analysis techniques can be employed to validate and refine the detected fire locations, minimizing false positives and improving overall reliability.

Overall, the architecture of forest fire detection using CNN-RF and CNN-XGBoost combines the strengths of deep learning and ensemble learning approaches to achieve accurate and efficient detection of forest fires. By integrating these techniques into a cohesive framework, the system can effectively analyze large-scale image data and provide timely alerts to mitigate the devastating effects of forest fires.

7.2 ALGORITHM

7.2.1 CNN-RF

In the realm of forest fire detection, the integration of Convolutional Neural Networks (CNNs) with Random Forest (RF) classifiers has emerged as a powerful approach to accurately identify and predict fire occurrences. This fusion leverages the strengths of both deep learning and ensemble learning techniques, offering a robust solution to the complex problem of forest fire detection.

At the heart of this architecture lies the CNN, a deep learning model specifically designed for image processing tasks. The CNN automatically extracts meaningful features from input images of forested areas, capturing spatial patterns, textures, and structures indicative of fire presence. By leveraging its hierarchical architecture of convolutional and pooling layers, the CNN can effectively learn discriminative features from raw image data.

Following feature extraction by the CNN, the extracted features serve as input to the RF classifier. The RF classifier, a type of ensemble learning algorithm, aggregates predictions from multiple decision trees trained on different subsets of the input features. This ensemble approach enhances the model's generalization ability and robustness against overfitting, resulting in more reliable predictions of forest fire occurrences.

By combining the feature extraction capabilities of CNNs with the ensemble learning prowess of RF classifiers, the CNN-RF architecture excels in detecting forest fires with high accuracy and efficiency. This integrated approach not only enhances the detection performance but also enables scalability to handle large-scale image datasets, thereby contributing to the timely detection and mitigation of forest fire disasters.

7.2.2 CNN-XGBOOST

In forest fire detection systems, the utilization of Convolutional Neural Networks (CNNs) in conjunction with Extreme Gradient Boosting (XGBoost) classifiers has emerged as a potent methodology for accurately identifying and predicting fire occurrences. This fusion capitalizes on the strengths of both deep learning and ensemble learning techniques, presenting a robust solution to the intricate challenge of forest fire detection.

At the core of this architecture lies the CNN, a specialized deep learning model adept at processing image data. The CNN autonomously extracts salient features from input images of forested areas, discerning spatial patterns, textures, and configurations indicative of fire presence. Leveraging its hierarchical structure of convolutional and pooling layers, the CNN effectively learns discriminative representations from raw image data, facilitating precise detection of fire-related features.

Following feature extraction by the CNN, the extracted features are fed into the XGBoost classifier. XGBoost, a variant of gradient boosting algorithms, iteratively constructs an ensemble of decision trees to optimize predictive performance. By sequentially refining predictions and minimizing the loss function, XGBoost enhances the model's ability to discern subtle patterns and variations in the input data, thereby improving the accuracy of fire detection.

By amalgamating the feature extraction capabilities of CNNs with the predictive power of XGBoost classifiers, the CNN-XGBoost architecture excels in forest fire detection with exceptional accuracy and efficiency. This integrated approach not only enhances detection performance but also facilitates scalability to handle vast amounts of image data, contributing significantly to the timely detection and mitigation of forest fire disasters.

7.4 SOURCE CODE

```
import os
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout, BatchNormalization
from PIL import Image
import numpy as np
import pandas as pd
import cv2
import glob
from tqdm import tqdm
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('dark_background')
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from tkinter import filedialo
#==================PREDICTION METRICS==============#
import tensorflow as tf
import keras.backend as K
def f1_score(y_true, y_pred): #taken from old keras source code
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
    precision = true_positives / (predicted_positives + K.epsilon())
    recall = true_positives / (possible_positives + K.epsilon())
    f1_val = 2*(precision*recall)/(precision+recall+K.epsilon())
    return f1_val
METRICS = [
      tf.keras.metrics.BinaryAccuracy(name='accuracy'),
      tf.keras.metrics.Precision(name='precision'),
      tf.keras.metrics.Recall(name='recall'),
      tf.keras.metrics.AUC(name='auc'),
        f1_score, ]
#=============METRICS VISUALISATION===============#
def
Train_Val_Plot(acc,val_acc,loss,val_loss,auc,val_auc,precision,val_precisi
on,f1,val_f1,recall,val_recall)
    fig, (ax1, ax2,ax3,ax4,ax5,ax6) = plt.subplots(1,6, figsize= (20,5))
    fig.suptitle(" MODEL'S METRICS VISUALIZATION ")
    ax1.plot(range(1, len(acc) + 1), acc)
    ax1.plot(range(1, len(val_acc) + 1), val_acc)
    ax1.set_title('History of Accuracy')
    ax1.set_xlabel('Epochs')
    ax1.set_ylabel('Accuracy')
    ax1.legend(['training', 'validation'])
    ax2.plot(range(1, len(loss) + 1), loss)
    ax2.plot(range(1, len(val_loss) + 1), val_loss)
    ax2.set_title('History of Loss')
    ax2.set_xlabel('Epochs')
    ax2.set_ylabel('Loss')
    ax2.legend(['training', 'validation'])
```

```
ax3.plot(range(1, len(auc) + 1), auc)
ax3.plot(range(1, len(val_auc) + 1), val_auc)
ax3.set_title('History of AUC')
ax3.set_xlabel('Epochs')
ax3.set_ylabel('AUC')
ax3.legend(['training', 'validation'])
ax4.plot(range(1, len(precision) + 1), precision)
ax4.plot(range(1, len(val_precision) + 1), val_precision)
ax4.set_title('History of Precision')
ax4.set_xlabel('Epochs')
ax4.set_ylabel('Precision')
ax4.legend(['training', 'validation'])
ax5.plot(range(1, len(f1) + 1), f1)
ax5.plot(range(1, len(val_f1) + 1), val_f1)
ax5.set_title('History of F1-score')
ax5.set_xlabel('Epochs')
ax5.set_ylabel('F1 score')
ax5.legend(['training', 'validation'])
ax6.plot(range(1, len(recall) + 1), recall)
ax6.plot(range(1, len(val_recall) + 1), val_recall)
ax6.set_title('History of Recall')
ax6.set_xlabel('Epochs')
ax6.set_ylabel('Recall')
ax6.legend(['training', 'validation'])
plt.show()
```

```
data = r'fire'
fire_img = glob.glob(r'fire\fired\*.png')
non_fire_img = glob.glob(r'fire\nonfired\*.png')
non_fire_img
fire_img
print('Number of images with fire : {}'.format(len(fire_img)))
print('Number of images with nonfire : {}'.format(len(non_fire_img)))
categories = ['fired', 'nonfired']
len_categories = len(categories)
print(len_categories)
image_count = {}
train_data = []
for i , category in tqdm(enumerate(categories)):
    class_folder = os.path.join(data, category)
    label = category
    image_count[category] = []
    for path in os.listdir(os.path.join(class_folder)):
        image_count[category].append(category)
        train_data.append(['{}/{}'.format(category, path), i, category])
#show image count
for key, value in image_count.items():
    print('{0} -> {1}'.format(key, len(value)))
#create a dataframe
df = pd.DataFrame(train_data, columns=['file', 'id', 'label'])
df.shape
df.head(def create_mask_for_plant(image):
    image_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_hsv = np.array([0,0,250])
    upper_hsv = np.array([250,255,255])
    mask = cv2.inRange(image_hsv, lower_hsv, upper_hsv)
    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (11,11))
    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
    return mask
#image segmentation function
def segment_image(image):
    mask = create_mask_for_plant(image)
    output = cv2.bitwise_and(image, image, mask = mask)
    return output/255
#sharpen the image
def sharpen_image(image):
    image_blurred = cv2.GaussianBlur(image, (0, 0), 3)
    image_sharp = cv2.addWeighted(image, 1.5, image_blurred, -0.5, 0)
    return image_sharp
from keras.preprocessing import image
# function to get an image
def read_img(filepath, size):
    img = image.load_img(os.path.join(data, filepath), target_size=size)
    #convert image to array
    img = image.img_to_array(img)
    return img
nb_rows = 3
nb_cols = 5
```

```
fig, axs = plt.subplots(nb_rows, nb_cols, figsize=(10, 5))
plt.suptitle('SAMPLE IMAGES')
for i in range(0, nb_rows):
    for j in range(0, nb_cols):
        axs[i, j].xaxis.set_ticklabels([])
        axs[i, j].yaxis.set_ticklabels([])
        axs[i, j].imshow((read_img(df['file'][np.random.randint(120)],
(255,255)))/255.)
plt.show()
#get an image
img = read_img(df['file'][80],(255,255))
#mask
image_mask = create_mask_for_plant(img)
#segmentation
image_segmented = segment_image(img)
#sharpen the image
image_sharpen = sharpen_image(image_segmented)
fig, ax = plt.subplots(1, 4, figsize=(10, 5));
plt.suptitle('SAMPLE PROCESSED IMAGE', x=0.5, y=0.8)
plt.tight_layout(1)
ax[0].set_title('ORIG.', fontsize=12)
ax[1].set_title('MASK', fontsize=12)
ax[2].set_title('SEGMENTED', fontsize=12)
ax[3].set_title('SHARPEN', fontsize=12
ax[0].imshow(img/255);
ax[1].imshow(image_mask);
ax[2].imshow(image_segmented);
```

```
ax[3].imshow(image_sharpen);
plt.show()
plt.figure(figsize = (10,10))
sns.countplot(x = "label",data = df)
plt.title("Non Fire vs Fire")
plt.show()
encoder = OneHotEncoder()
encoder.fit([[0], [1]])
data = []
paths = []
result = []
for r, d, f in os.walk(r'fire/fired'):
    for file in f:
        if '.png' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = Image.open(path)
    img = img.resize((65,65))
    img = np.array(img)
    if(img.shape == (65,65,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[0]]).toarray())
# This cell updates result list for images without tumor
paths = []
for r, d, f in os.walk(r"fire/nonfired"):
    for file in f:
        if '.png' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = Image.open(path)
    img = img.resize((65,65))
    img = np.array(img)
    if(img.shape == (65,65,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[1]]).toarray())
data = np.array(data)
print(data.shape)
result = np.array(result)
result = result.reshape(118,2)
x_train,x_test,y_train,y_test = train_test_split(data, result, test_size=0.2,
shuffle=True, random_state=0)
#cnn
model = Sequential()
model.add(Conv2D(32, kernel_size=(2, 2), input_shape=(65, 65, 3),
padding = 'Same'))
model.add(Conv2D(32, kernel_size=(2, 2),  activation ='relu', padding =
'Same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding =
'Same'))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding =
'Same')
```

```
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))
model.compile(loss = "categorical_crossentropy", optimizer='Adamax',
metrics=METRICS)
print(model.summary())
history = model.fit(x_train, y_train,validation_data=(x_test,y_test), epochs
= 5, batch_size = 5, verbose = 1)
cnn=model.evaluate(x_test,y_test,verbose=1)[1]*100
print('CNN accuracy is:',cnn,'%')
Train_Val_Plot(history.history['accuracy'],history.history['val_accuracy'],
               history.history['loss'],history.history['val_loss'],
               history.history['auc'],history.history['val_auc'],
               history.history['precision'],history.history['val_precision'],
               history.history['f1_score'],history.history['val_f1_score'],
               history.history['recall'],history.history['val_recall']
              )
#model.save('model.h5')
#from keras.models import load_model
#model = load_model('model.h5')
def names(number):
    if number==0:
        return 'Fire'
```

```
else:
        return 'Non-Fire'
from matplotlib.pyplot import imshow
from tkinter import filedialog
import PIL
Image = filedialog.askopenfilename()
img = PIL.Image.open(Image)
x = np.array(img.resize((65,65)))
x = x.reshape(1,65,65,3)
res = model.predict_on_batch(x)
classification = np.where(res == np.amax(res))[1][0]
imshow(img)
print(str(res[0][classification]*100) + '% Confidence This Is ' +
names(classification))
```

8. SYSTEM TESTING

8.1 Types of Testing

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. The candidate system is subject to variety of tests-on-line response, Volume Street, recovery and security and usability test. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

UNIT TESTING

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

INTEGRATION TESTING

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing.

ii) Bottom-up integration testing.

WHITE BOX TESTING

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we derived test cases that guarantee that all independent paths within a module have been exercised at least once.

BLACK BOX TESTING

- ✓ Black box testing is done to find incorrect or missing function
- ✓ Interface error
- ✓ Errors in external database access
- ✓ Performance errors
- ✓ Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

VALIDATION TESTING

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

USER ACCEPTANCE TESTING

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

OUTPUT TESTING

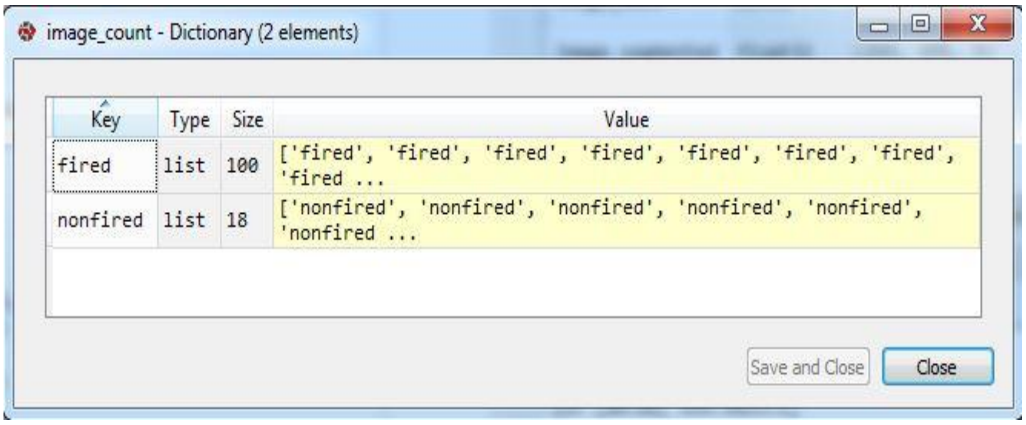
After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

8.2 TEST CASES

S.NO	Input	Expected Output
1		Non-Fire
2		Fire
3		Non-Fire
4		Fire

9. OUTPUT SCREENS

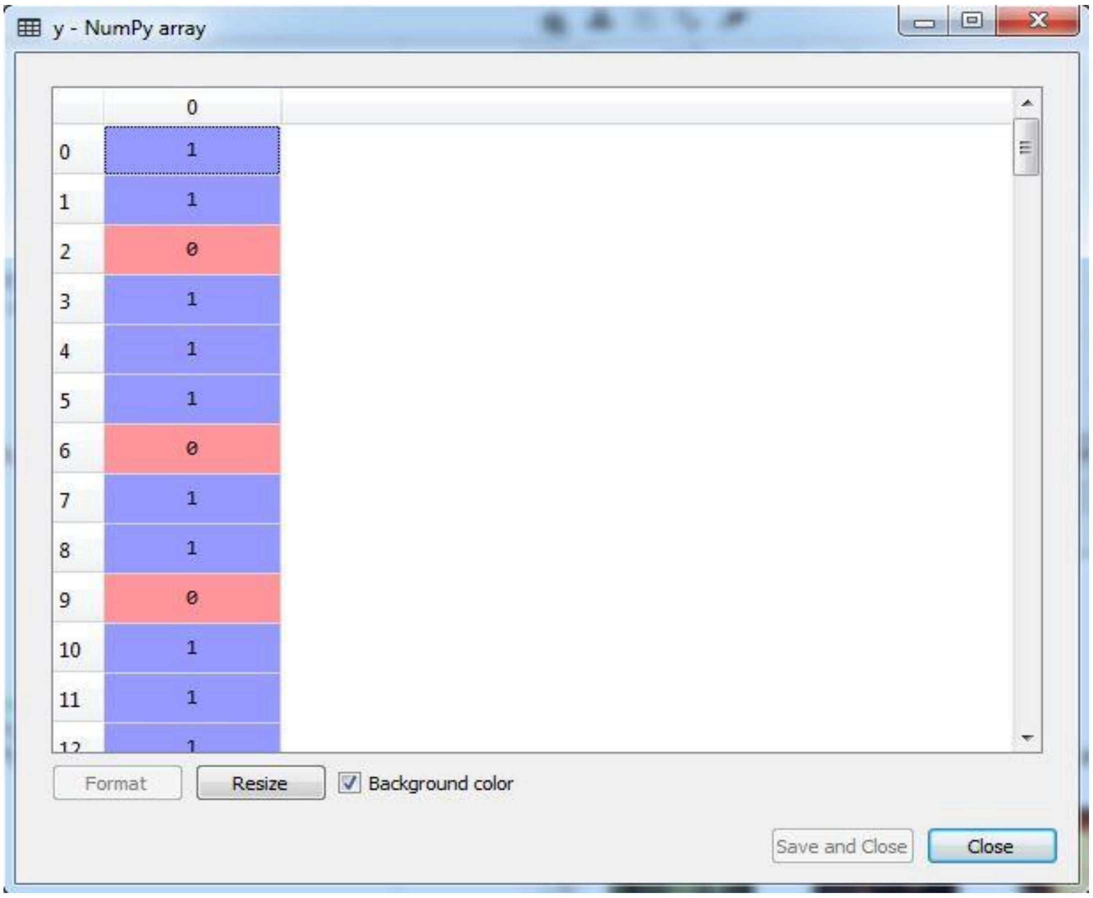
9.1 Splitting of the Image Dataset Screen


image_count - Dictionary (2 elements)

Key	Type	Size	Value
fired	list	100	['fired', 'fired', 'fired', 'fired', 'fired', 'fired', 'fired', 'fired ...
nonfired	list	18	['nonfired', 'nonfired', 'nonfired', 'nonfired', 'nonfired', 'nonfired ...

FIGURE:9.1 Splitting of the Image Dataset Screen

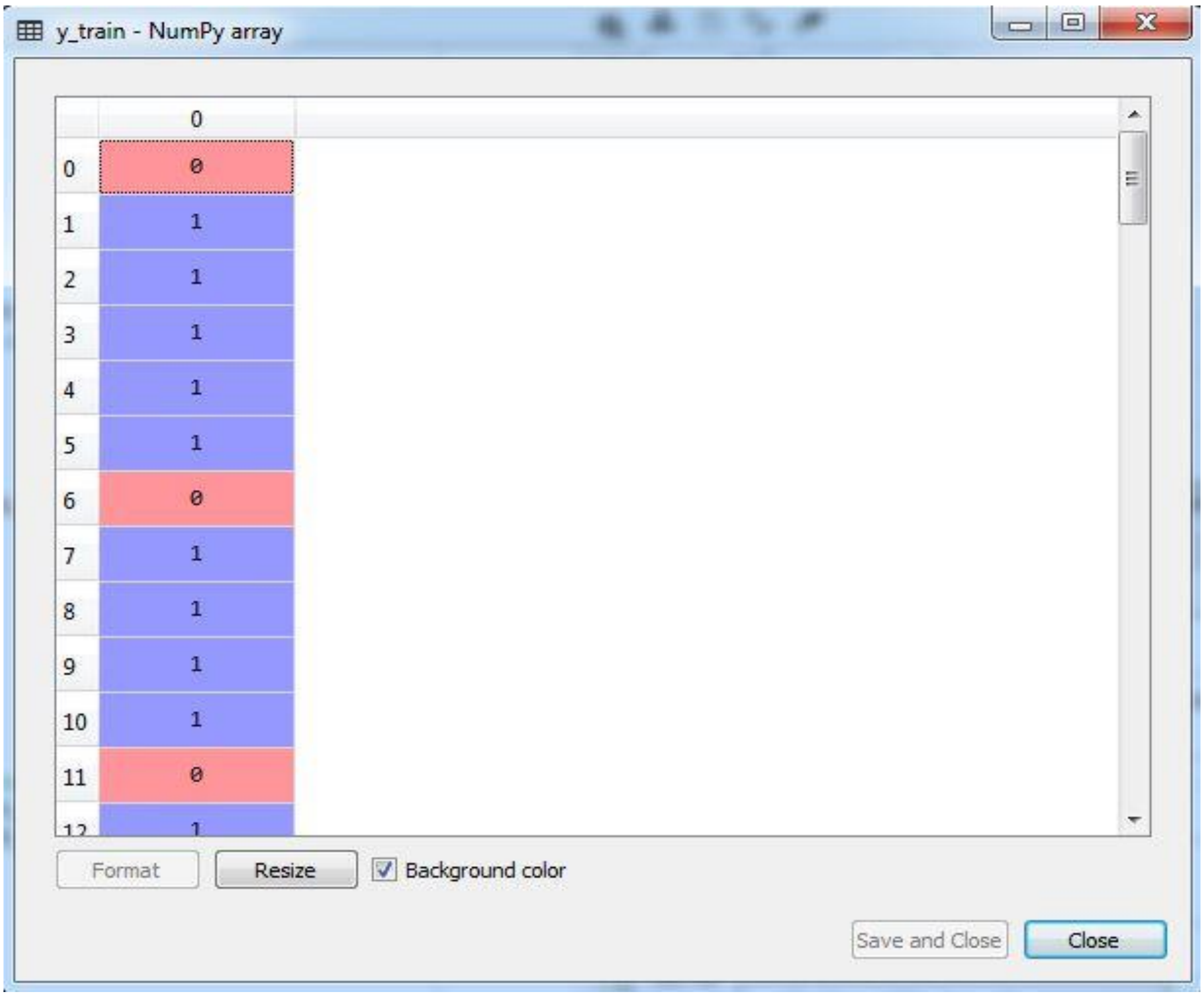
9.2 DATA SET


y - NumPy array

	0
0	1
1	1
2	0
3	1
4	1
5	1
6	0
7	1
8	1
9	0
10	1
11	1
12	1

FIGURE :9.2 DATA SET

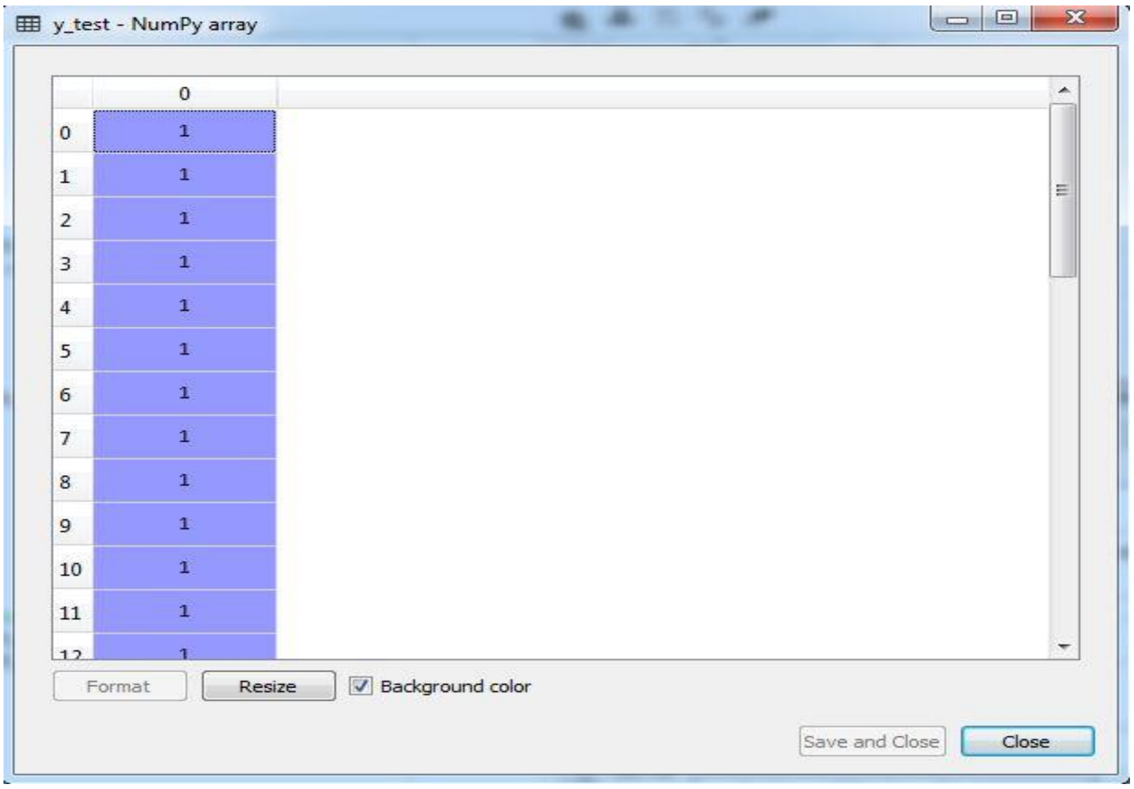
9.3 TRAINING DATASET



	0
0	0
1	1
2	1
3	1
4	1
5	1
6	0
7	1
8	1
9	1
10	1
11	0
12	1

FIGURE:9.3 TRAINING DATASET

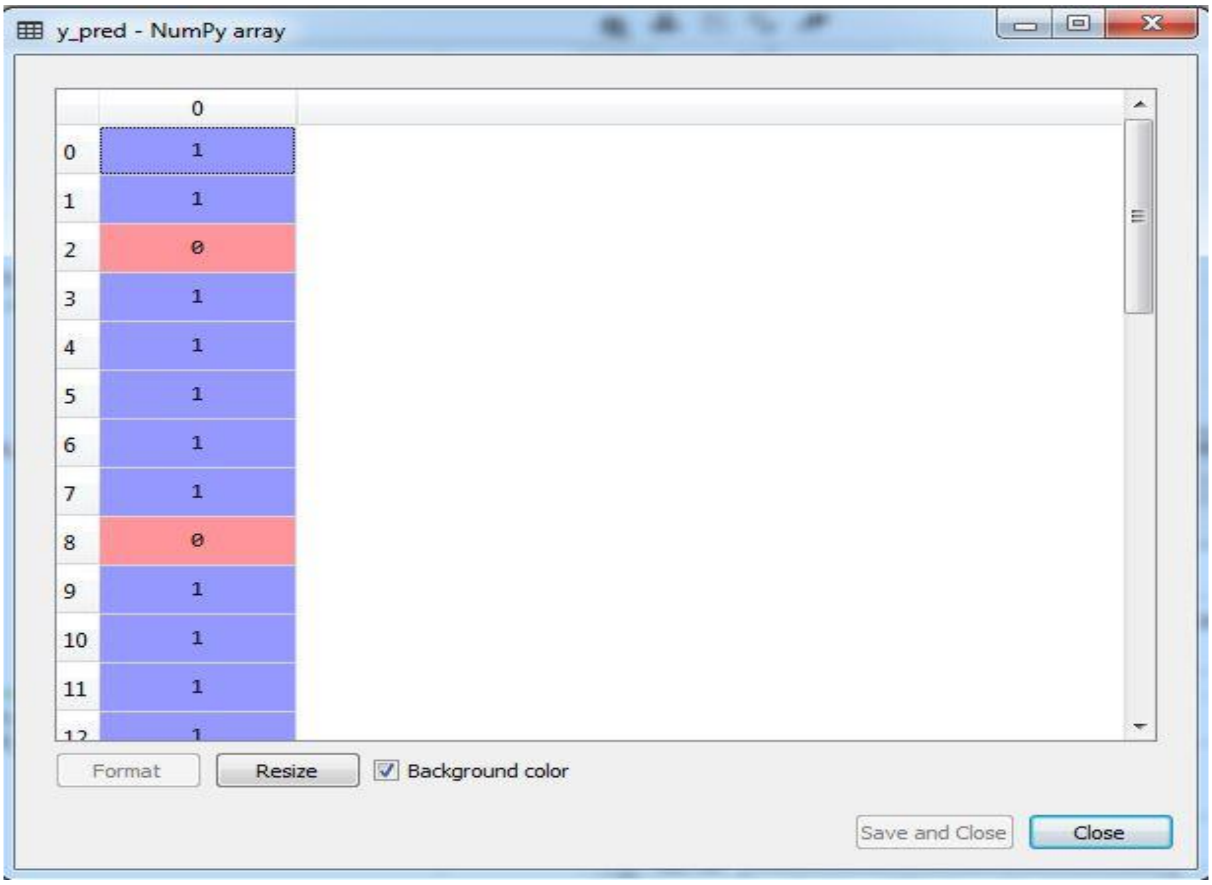
9.4 TESTING DATASET



	0
0	1
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1

FIGURE:9.4 TESTING DATASET

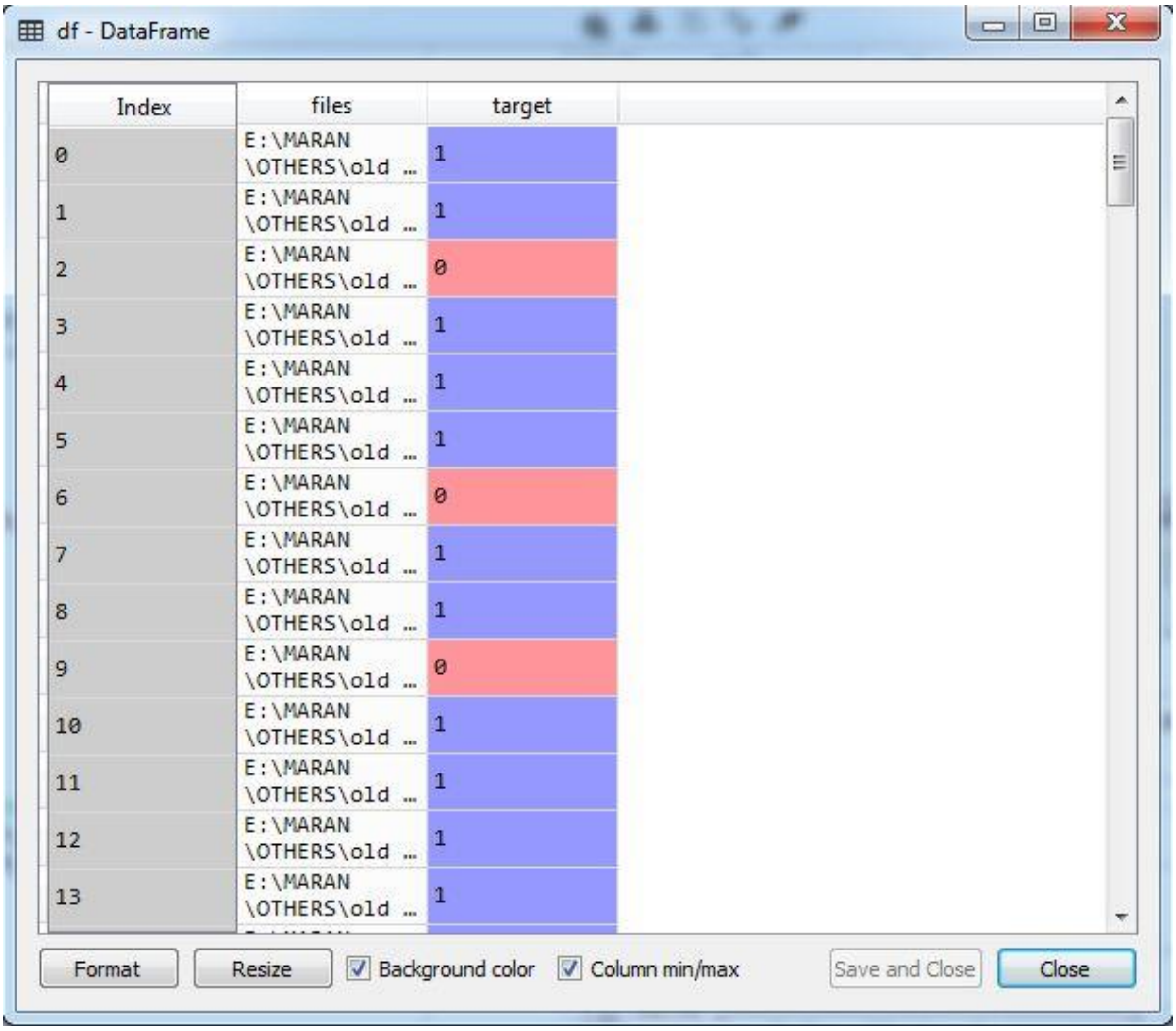
9.5 PREDICTION


y_pred - NumPy array

	0
0	1
1	1
2	0
3	1
4	1
5	1
6	1
7	1
8	0
9	1
10	1
11	1
12	1

FIGURE:9.5 PREDICTION

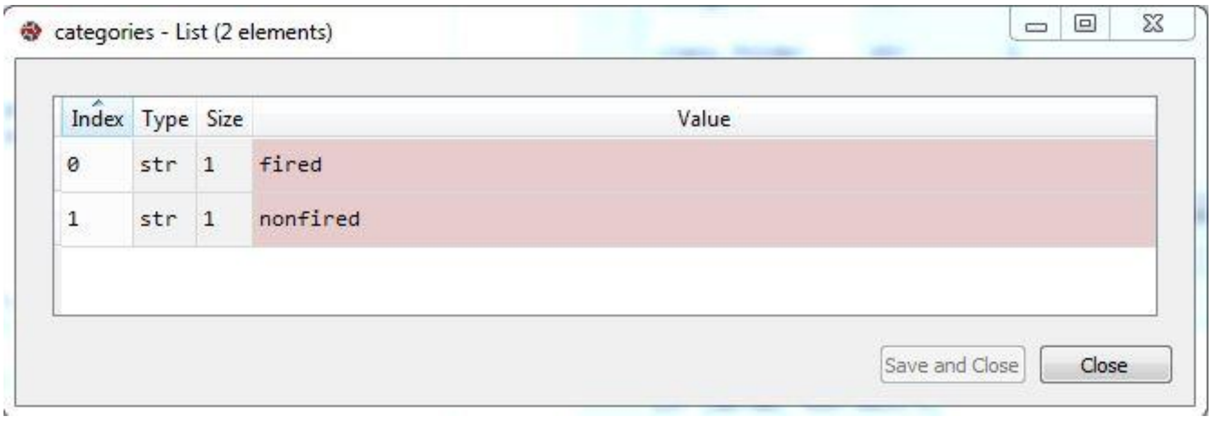
9.6 DATAFRAME



df - DataFrame

Index	files	target
0	E:\MARAN \OTHERS\old …	1
1	E:\MARAN \OTHERS\old …	1
2	E:\MARAN \OTHERS\old …	0
3	E:\MARAN \OTHERS\old …	1
4	E:\MARAN \OTHERS\old …	1
5	E:\MARAN \OTHERS\old …	1
6	E:\MARAN \OTHERS\old …	0
7	E:\MARAN \OTHERS\old …	1
8	E:\MARAN \OTHERS\old …	1
9	E:\MARAN \OTHERS\old …	0
10	E:\MARAN \OTHERS\old …	1
11	E:\MARAN \OTHERS\old …	1
12	E:\MARAN \OTHERS\old …	1
13	E:\MARAN \OTHERS\old …	1

FIGURE:9.6 DATAFRAME



categories - List (2 elements)

Index	Type	Size	Value
0	str	1	fired
1	str	1	nonfired

FIGURE:9.7 CATEGORIES OF THE DATA SET

```
Number of images with fire : 100
Number of images with nonfire : 18
2
2it [00:00, 666.61it/s]
fired -> 100
nonfired -> 18
```

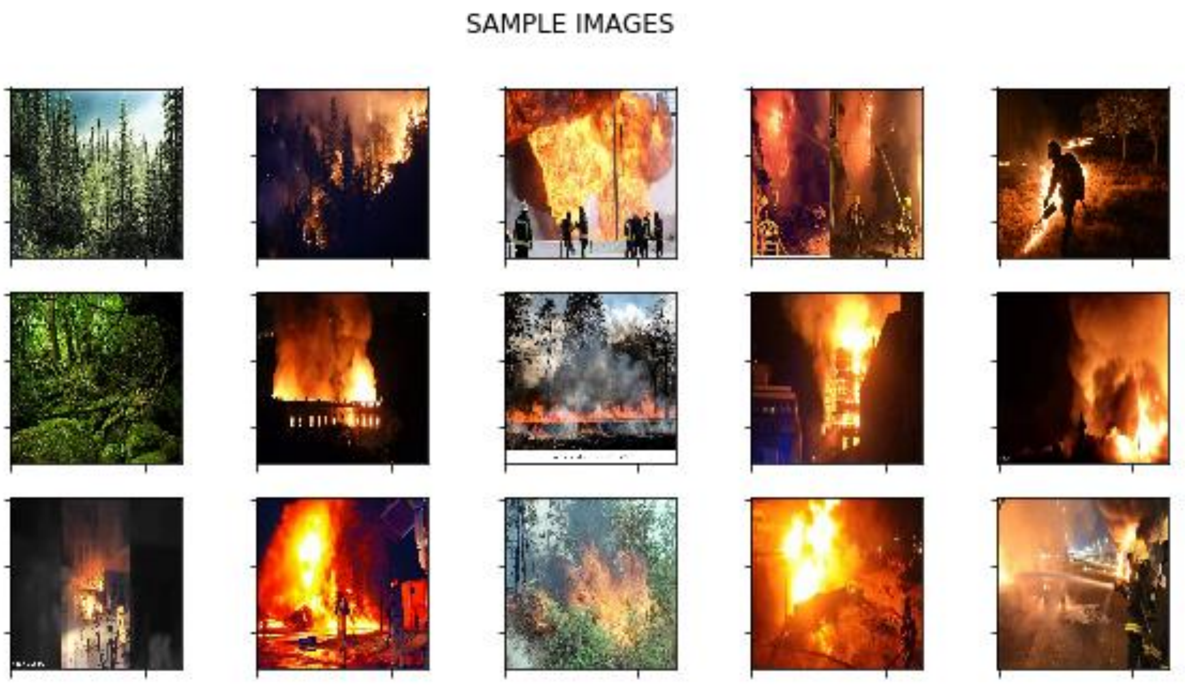


FIGURE:9.8 SAMPLE IMAGES

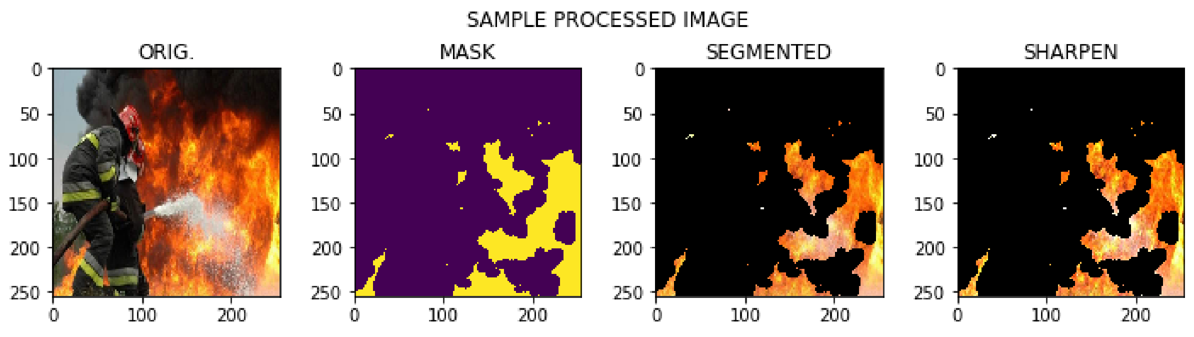


FIGURE:9.9 PROCESSED IMAGE

9.10 BAR GRAPH BETWEEN NON FIRE AND FIRE IMAGES

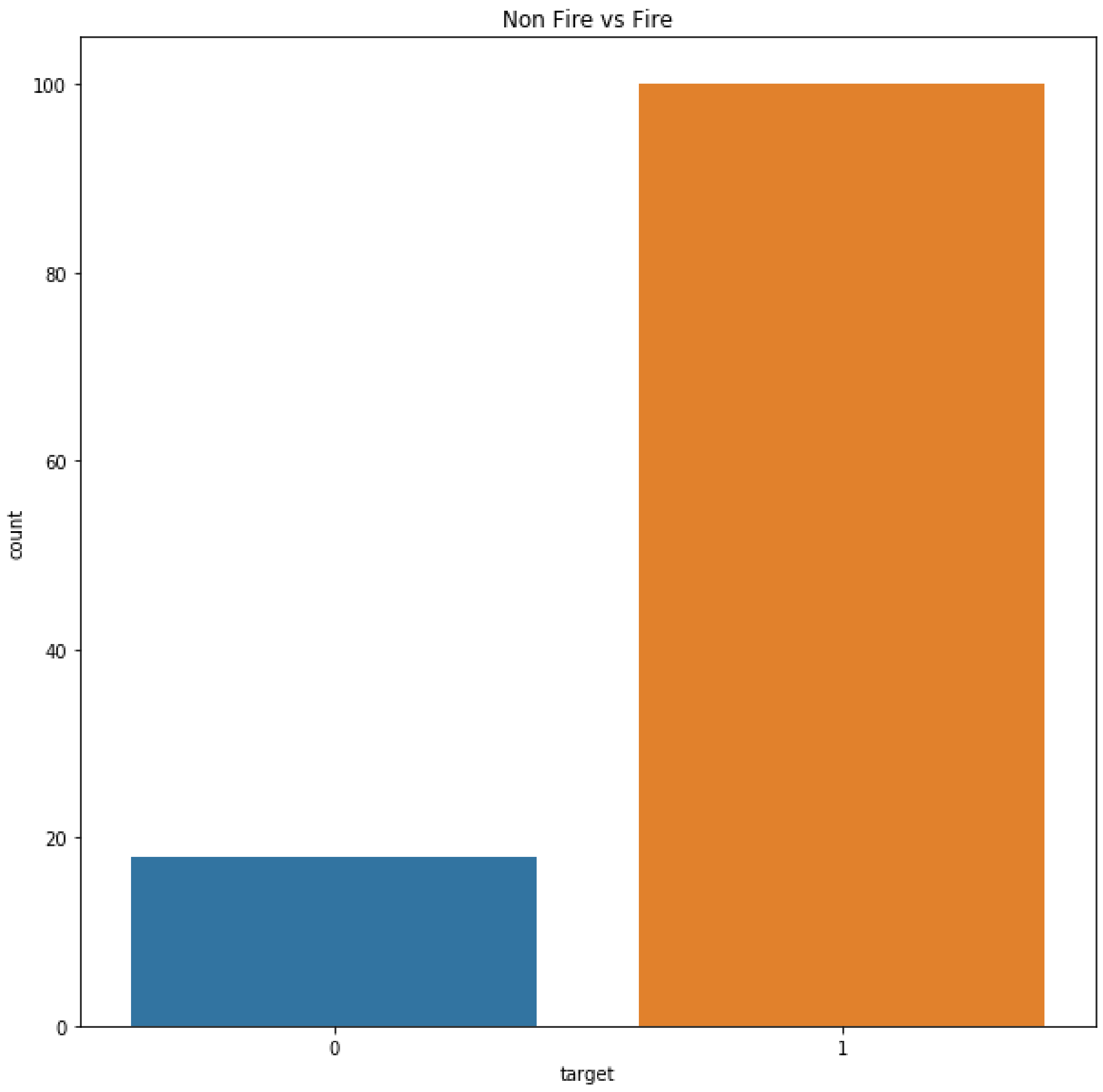
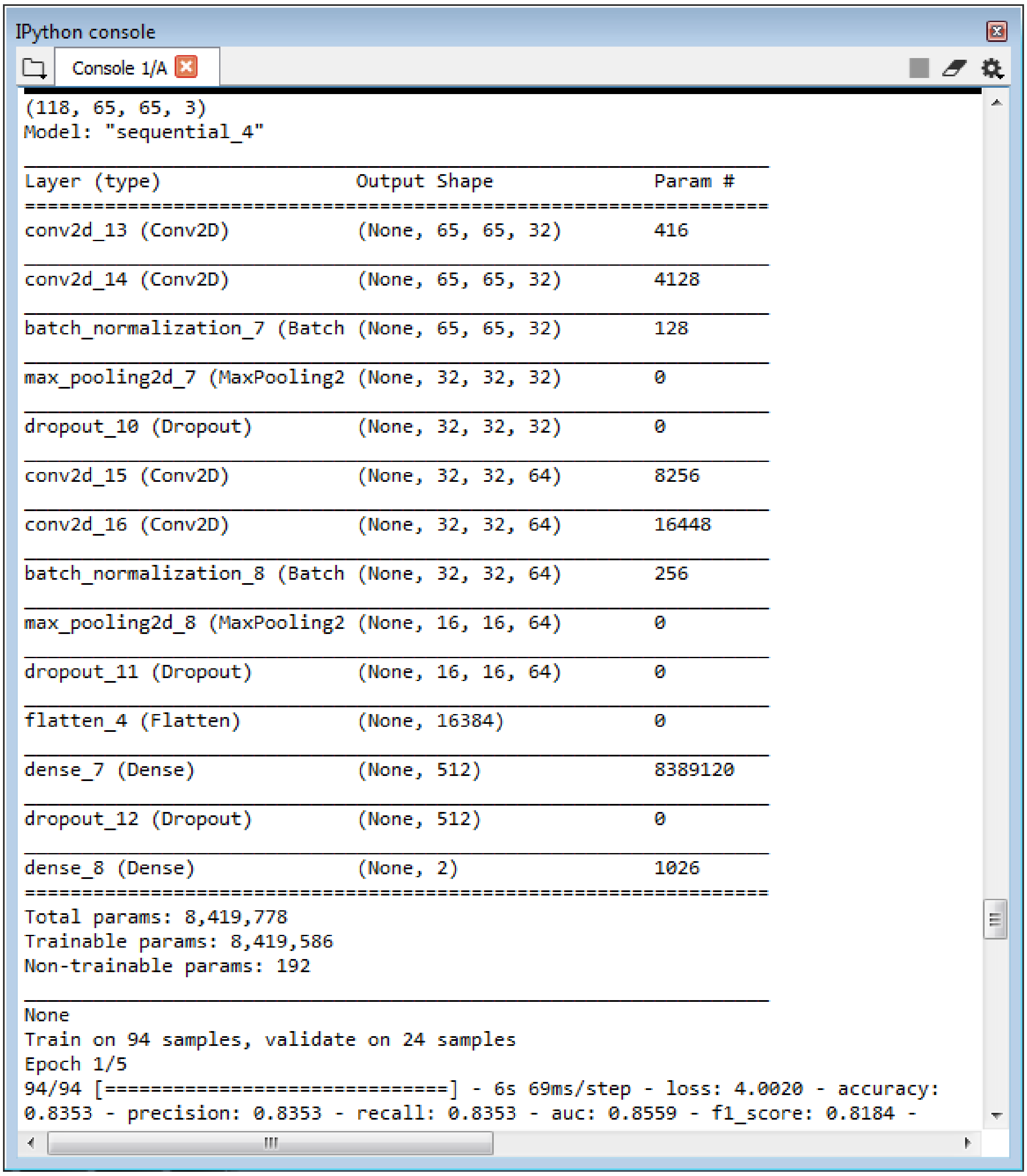


FIGURE:9.10 BAR GRAPH BETWEEN NON FIRE AND FIRE IMAGES

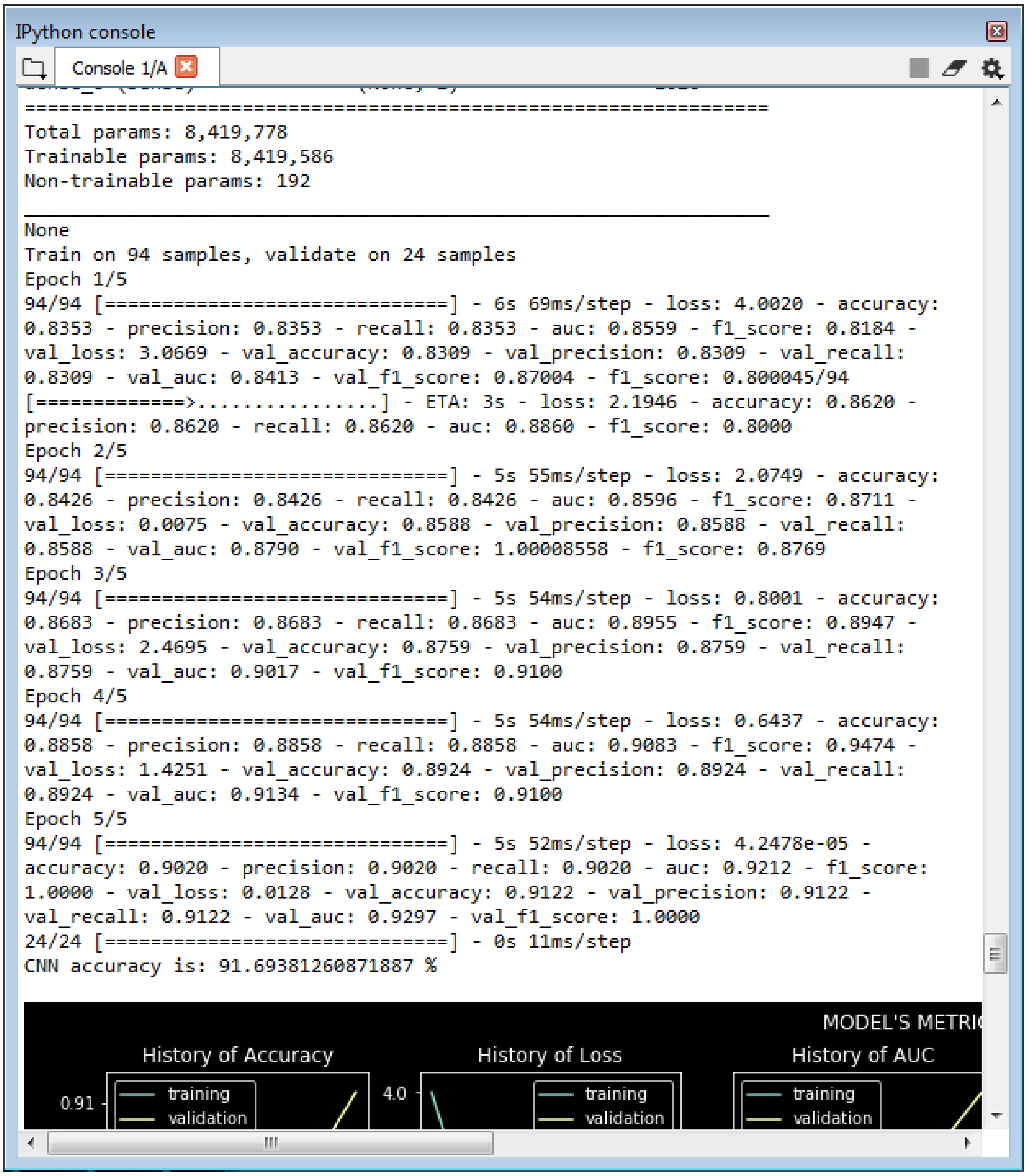
9.11 CNN-RF AND CNN-XGBOOST PARAMETERS


IPython console

Console 1/A

```
(118, 65, 65, 3)
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_13 (Conv2D)           (None, 65, 65, 32)        416
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 65, 65, 32)        4128
_________________________________________________________________
batch_normalization_7 (Batch (None, 65, 65, 32)        128
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 32, 32, 32)        0
_________________________________________________________________
dropout_10 (Dropout)         (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 32, 32, 64)        8256
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 32, 32, 64)        16448
_________________________________________________________________
batch_normalization_8 (Batch (None, 32, 32, 64)        256
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 16, 16, 64)        0
_________________________________________________________________
dropout_11 (Dropout)         (None, 16, 16, 64)        0
_________________________________________________________________
flatten_4 (Flatten)          (None, 16384)             0
_________________________________________________________________
dense_7 (Dense)              (None, 512)               8389120
_________________________________________________________________
dropout_12 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_8 (Dense)              (None, 2)                 1026
=================================================================
Total params: 8,419,778
Trainable params: 8,419,586
Non-trainable params: 192
_________________________________________________________________
None
Train on 94 samples, validate on 24 samples
Epoch 1/5
94/94 [==============================] - 6s 69ms/step - loss: 4.0020 - accuracy:
0.8353 - precision: 0.8353 - recall: 0.8353 - auc: 0.8559 - f1_score: 0.8184 -
```

FIGURE:9.11 CNN-RF AND CNN-XGBOOST PARAMETERS



```
===================================================================
Total params: 8,419,778
Trainable params: 8,419,586
Non-trainable params: 192

___________________________________________________________________
None
Train on 94 samples, validate on 24 samples
Epoch 1/5
94/94 [==============================] - 6s 69ms/step - loss: 4.0020 - accuracy:
0.8353 - precision: 0.8353 - recall: 0.8353 - auc: 0.8559 - f1_score: 0.8184 -
val_loss: 3.0669 - val_accuracy: 0.8309 - val_precision: 0.8309 - val_recall:
0.8309 - val_auc: 0.8413 - val_f1_score: 0.87004 - f1_score: 0.800045/94
[============>................] - ETA: 3s - loss: 2.1946 - accuracy: 0.8620 -
precision: 0.8620 - recall: 0.8620 - auc: 0.8860 - f1_score: 0.8000
Epoch 2/5
94/94 [==============================] - 5s 55ms/step - loss: 2.0749 - accuracy:
0.8426 - precision: 0.8426 - recall: 0.8426 - auc: 0.8596 - f1_score: 0.8711 -
val_loss: 0.0075 - val_accuracy: 0.8588 - val_precision: 0.8588 - val_recall:
0.8588 - val_auc: 0.8790 - val_f1_score: 1.00008558 - f1_score: 0.8769
Epoch 3/5
94/94 [==============================] - 5s 54ms/step - loss: 0.8001 - accuracy:
0.8683 - precision: 0.8683 - recall: 0.8683 - auc: 0.8955 - f1_score: 0.8947 -
val_loss: 2.4695 - val_accuracy: 0.8759 - val_precision: 0.8759 - val_recall:
0.8759 - val_auc: 0.9017 - val_f1_score: 0.9100
Epoch 4/5
94/94 [==============================] - 5s 54ms/step - loss: 0.6437 - accuracy:
0.8858 - precision: 0.8858 - recall: 0.8858 - auc: 0.9083 - f1_score: 0.9474 -
val_loss: 1.4251 - val_accuracy: 0.8924 - val_precision: 0.8924 - val_recall:
0.8924 - val_auc: 0.9134 - val_f1_score: 0.9100
Epoch 5/5
94/94 [==============================] - 5s 52ms/step - loss: 4.2478e-05 -
accuracy: 0.9020 - precision: 0.9020 - recall: 0.9020 - auc: 0.9212 - f1_score:
1.0000 - val_loss: 0.0128 - val_accuracy: 0.9122 - val_precision: 0.9122 -
val_recall: 0.9122 - val_auc: 0.9297 - val_f1_score: 1.0000
24/24 [==============================] - 0s 11ms/step
CNN accuracy is: 91.69381260871887 %
```

FIGURE:9.12 CNN-RF AND CNN-XGBOOST PARAMETERS

9.13 MODEL METRICS VISUALIZATION

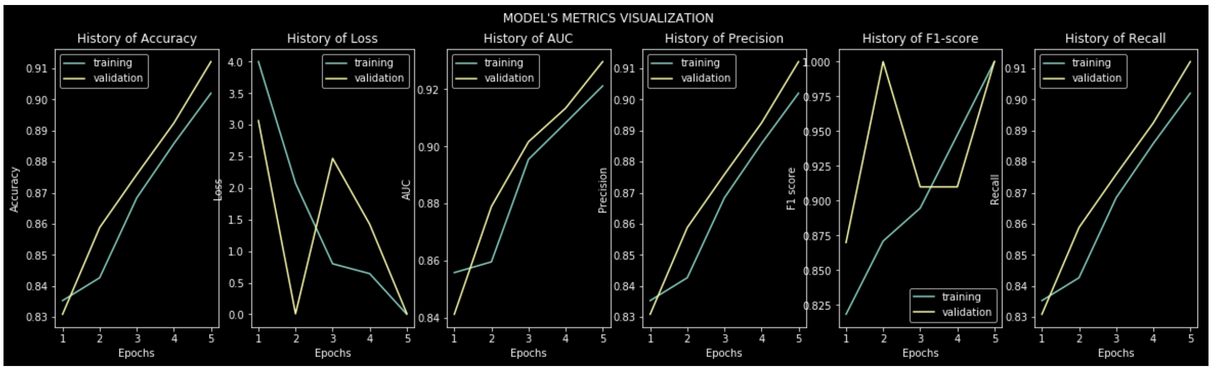
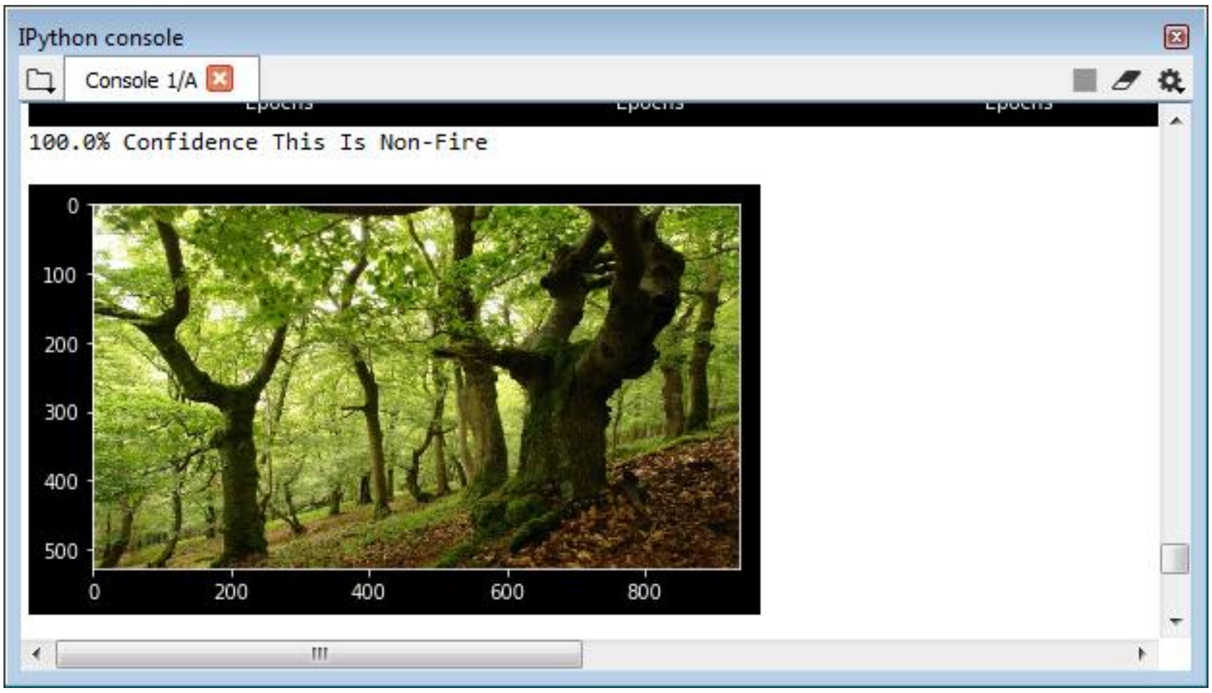


FIGURE:9.13 MODEL METRICS VISUALIZATION

9.14 FINAL OUTPUT



```
100.0% Confidence This Is Non-Fire
```

FIGURE:9.14 FINAL OUTPUT

10. CONCLUSION

Forest fire detection using CNN-RF and CNN-XGBoost represents a cutting-edge approach that combines the strengths of deep learning and ensemble learning techniques. Through the integration of Convolutional Neural Networks (CNNs) with Random Forest (RF) and Extreme Gradient Boosting (XGBoost) classifiers, this methodology offers a robust and effective solution to the critical challenge of early detection and prediction of forest fires.

In conclusion, the CNN-RF and CNN-XGBoost architectures demonstrate remarkable capabilities in forest fire detection. These methodologies leverage CNNs' ability to automatically extract relevant features from raw image data, capturing intricate spatial patterns and textures indicative of fire presence. By integrating ensemble learning techniques such as RF and XGBoost, the models enhance predictive performance, robustness, and generalization ability, thereby improving the accuracy of fire detection.

Furthermore, forest fire detection using CNN-RF and CNN-XGBoost architectures offers several key advantages. Firstly, it enables timely detection of fire occurrences, facilitating prompt response and mitigation efforts to minimize the devastating impacts of forest fires on ecosystems, livelihoods, and infrastructure. Secondly, the integration of deep learning and ensemble learning techniques enhances the scalability and adaptability of the detection system, allowing it to handle large-scale image datasets and varying environmental conditions. Additionally, the models exhibit high accuracy and reliability in detecting forest fires, reducing false positives and false negatives compared to traditional detection methods.

Moreover, the CNN-RF and CNN-XGBoost architectures hold promise for future advancements in forest fire detection technology. Continued research and development efforts can further refine these methodologies, exploring novel techniques for feature extraction, model optimization, and real-time monitoring. Additionally, the integration of additional data sources such as meteorological data, satellite imagery, and geographical information can enhance the predictive capabilities of the models, enabling more comprehensive and proactive fire management strategies.

In summary, forest fire detection using CNN-RF and CNN-XGBoost architectures represents a significant advancement in the field of wildfire management. By leveraging the synergy between deep learning and ensemble learning techniques, these methodologies offer a potent tool for safeguarding ecosystems, communities, and resources from the destructive impact of forest fires.

11. FURTHER ENHANCEMENTS / RECOMMENDATIONS

To further enhance forest fire detection using CNN-RF and CNN-XGBoost, researchers could explore the integration of multi-modal data sources. Combining satellite imagery with ground-based sensor data, weather forecasts, and historical fire occurrence records can provide a more comprehensive understanding of fire dynamics and improve prediction accuracy. Additionally, incorporating real-time data streams and IoT (Internet of Things) sensors into the detection system can enable timely updates and adaptive response strategies, enhancing the system's effectiveness in dynamic fire environments.

Furthermore, advancements in model interpretability techniques could facilitate better understanding and trust in the detection system's decision-making process. Techniques such as feature importance analysis, SHAP (SHapley Additive exPlanations), and model visualization tools can help elucidate the underlying factors driving fire predictions, enabling stakeholders to make informed decisions and prioritize resources effectively. Moreover, ongoing research into anomaly detection algorithms and outlier analysis methods could enhance the detection system's capability to identify emerging fire risks and anomalies in complex environmental conditions, enabling proactive fire management strategies.

12. REFERENCES

[1] Jamil Ahmad, Khan Muhammad, (2018) “Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications” in IEEE Trans on Systems, Man, and cybernetics: systems, Volume. 49, no. 7, pages. 1419 – 1434.

[2] Panagiotis Barmpoutis, Kosmas Dimitropoulos, (2018) “Spatio-Temporal Flame Modeling and Dynamic Texture Analysis for Automatic Video-Based Fire Detection” in IEEE Trans on Circuits and Systems for Video Technology, Volume. 25, No. 2, pages. 339 – 351,.

[3] Fang Chen, Zhengyang Lin, (2017) “Feng Yun-3C VIRR Active Fire Monitoring: Algorithm Description and Initial Assessment Using MODIS and Landsat Data” in IEEE Trans on geoscience and remote sensing, Volume. 55, no. 11, pages. 6420 – 6430,.

[4] Jun Wang, Thomas N. Polivka, (2018) “Improving Nocturnal Fire Detection with the VIIRS Day-Night Band” in IEEE Trans on geoscience and remote sensing, Volume. 54, no. 9, pages. 5503 – 5519,.

[5] Crispin Lobo, Yogesh Deshpande, (2018) “Forest Monitoring System Using Sensors, Wireless Communication, and Image Processing” in International Conference on Computing Communication Control and Automation,.

[6] Ju Jia Zou, Nargess Ghassempour, (2018) “A SIFT-Based Forest Fire Detection Framework Using Static Images” in International Conference on Signal Processing and Communication Systems,.

[7] Simon Jones, Samuel Hislop, (2018) “A New SemiAutomatic Seamless cloud-free Landsat Mosaicing Approach Tracks Forest Change Over Large Extents” in IEEE International Geoscience and Remote Sensing Symposium,.

[8] Fabiano Morelli, Alberto Setzer, (2018) “Land Use and Land Cover Dynamics to Fire Recurrence in the Brazilian Amazon” in IEEE International Geoscience and Remote Sensing Symposium,.

[9] Jin Fu, Jiye Qian, (2018) “Automatic Early Forest Fire Detection Based on Gaussian Mixture Model” in IEEE International Conference on Communication Technology,.

[10] Cristhian Bone, Diego Reyes, (2018) “Use of Multitemporal Indexes in the Identification of Forest Fires – A Case Study of Southern Chile” in IEEE,.

[11] Youmin Zhang, Yanhong Chen, (2019) “UAV Image-based Forest Fire Detection Approach Using Convolutional Neural Network” in IEEE Conference on Industrial Electronics and Applications,.

[12] Kiadtikornthaweeyot, Sukawattanavijit, (2018) “Automatic detection of forest fire burnt scar from Landsat-8 image of the northern part of Thailand” in International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology.

[13] Jae Kang Lee, Yong Huh, (2017) “Enhanced contextual forest fire detection with prediction interval analysis of surface temperature using vegetation Amount” in International journal of remote sensing, Volume. 38, No. 11, pages. 3375–3393,.

[14] D. P. Roy, S. S. Kumar, (2017) “Global operational land imager Landsat-8 reflectance-based active fire detection algorithm” in International Journal Of Digital Earth, Volume. 11, No. 2, pages. 154-178,.

[15] Zhixiang Liu, Chi Yuan, (2017) “Fire Detection Using Infrared Images for UAV-based Forest Fire Surveillance” in International Conference on Unmanned Aircraft Systems,.

[16] B.S. Sathish, Ganesan P, (2016) “A Comparative Approach of Identification and Segmentation of Forest Fire Region in High-Resolution Satellite Images” in Conference on futuristic trends in Research and innovation for social welfare,.

[17] Khaled A. Ghamry, Chi Yuan, (2016) “Unmanned Aerial Vehicle-Based Forest Fire Monitoring and Detection Using Image Processing Technique” in IEEE Chinese Guidance, Navigation and Control Conference.