

Early Warning System for Forest Fires using Surveillance Drone

Megha Kolhekar¹, Payal Chatterjee², Minal Kale³, Suyash Mali⁴, Chetan Velonde⁵

¹Associate Professor, ^{2,3,4,5}UG-student

Fr. Conceicao Rodrigues Institute of Technology, Navi Mumbai, India, 400703,

chatterjeepayal2705@gmail.com

Received on: 29 March, 2023

Revised on: 17 April, 2023

Published on: 19 April, 2023

Abstract –Forest fires are one of the major reasons of concern in many regions in the world, causing widespread damage to both mankind and the ecosystem. The development of an efficient system for the early detection of forest fires is crucial to enable rapid response and minimize the damage caused. To address this issue, an early warning system for detection of forest fires has been proposed in this paper. The system comprises of a surveillance drone equipped with a First Person View (FPV) camera and a Machine Learning algorithm that can detect potential fire outbreaks in the forest. The Machine Learning algorithm with the help of Image processing and Computer vision technology, detects the occurrence of forest fire from the FPV camera video feed. If the algorithm detects a fire outbreak, an alert is sent to the forest rangers, allowing them to quickly respond and prevent the fire from spreading. The proposed system has the potential to revolutionize forest fire management by providing a reliable and efficient early warning system for forest fires.

Keywords-Forest fire, Convolution Neural Networks (CNN), Unmanned Aerial Vehicles (UAV), You Only Look Once (YOLO), Inception model, Flight controller mechanism.

I - INTRODUCTION

Wildfires have severe impacts on local ecosystems that may continue to affect the flora and fauna of the area

for many years. A real threat to people's lives can come from sudden, out-of-control wildfires. Forest fires differ from urban fires in that they are more difficult to contain and provide a greater threat to lives and property. Therefore, it is crucial to intervene in wildfires quickly and effectively. Regulation of forest fires is a crucial topic where UAVs can have a significant impact. Moreover, CNN have developed into an efficient tool for image classification tasks like identifying forest fires in satellite images or drone footages [11].

A. Problem Overview

The automatic fire detection system using video feed and thermal analysis is the most effective method for combating forest fires. Early detection, localization, and monitoring of forest fires are made possible by these types of technology. The detection mechanism used today is like watching towers, satellite imagery, video recording over long distances, etc. However, these do not provide a solution to improve the effectiveness for the detection of forest fire. Video type detection of forest fire can reduce the strain on manpower and improve efficiency. For the same purpose, CNN is used to detect fire from the obtained video feed [11].

A forest fire detection system that uses a CNN architecture, trained on a dataset that contains images of normal scenery and forest fires, to automatically detect and classify fires from video feeds obtained from aerial

surveillance vehicles is described in this technical paper. The CNN model is tailored to attain high accuracy and fast detection rates. It is trained on a huge dataset of annotated images of forest fires and non-fire images. This system comprises of a surveillance drone with a FPV camera installed on it. The effectiveness of the trained CNN model in detecting fires with high accuracy and low false positive rates will be demonstrated by testing its performance on a test dataset of actual images of forest fires.

Table 1- State wise Incidence of Forest Fires and Burnt Area (In Hectares) in India from 2003-2006 for dominant states [1]

State	Number of fire incidents during 2003-06	Burnt area (in hectares) during 2003-06
Andhra Pradesh	701	5,308
Chattisgarh	778	6,424
Gujurat	2,782	27,617
Himachal Pradesh	1,794	28,880
Karnataka	721	3,382
Kerala	1,885	19,924
Madhya Pradesh	2,535	36,565
Maharashtra	7,823	134,971
Punjab	892	15,147
Tamil Nadu	1,355	6,657
Uttar Pradesh	347	2,754
Uttaranchal	3,325	14,047
Mizoram	161	11,498
Tripura	948	9,460

II- LITERATURE REVIEW

Traditional fire detection systems rely on sensors that detect changes in temperature or smoke density to trigger alarms. However, these systems have limitations such as false alarms and slow response times.

In recent years, with the advances in computer vision and deep learning techniques, CNN have emerged as a

promising solution for detecting forest fires from visual data such as satellite images, aerial photographs, or ground-based cameras. The use of CNN for forest fire detection has gained significant attention from researchers and practitioners due to its accuracy and efficiency. CNN can learn to identify the patterns and features that are indicative of forest fires, such as smoke plumes, changes in color or texture, and heat signatures.

Numerous studies have been conducted to develop and evaluate CNN-based models for forest fire detection. These studies have explored different aspects of the problem, such as the choice of input data, model architectures, optimization techniques, and evaluation metrics. Some studies have focused on using traditional RGB images, while others have explored the use of multispectral images or combinations of different data sources. Model architectures range from simple CNNs to more complex architectures such as Faster R-CNN, YOLO [2], or Mask R-CNN. Optimization techniques such as transfer learning, data augmentation, and fine-tuning have also been explored. Evaluation metrics used in these studies include accuracy, precision, recall, and F1-score.

A. UAVs for disaster management

In disaster management, parameters such as the response time is very crucial to provide proper on-time relief to the people residing in the affected areas. One of the most efficient ways of providing disaster relief is achieved through air surveillance. One of the studies, implementing UAVs for disaster management implement network architectures for geophysical, climate-induced, and meteorological disasters based on interaction between the UAV and wireless sensor network. This survey of advances in unmanned aerial vehicles (UAVs) for network-assisted first response to disaster management covers disaster prediction, assessment, and response.

The study examines the most recent developments in network-assisted first aid for disaster management using UAVs and points out unresolved problems. It provides a method for categorizing disasters in specific, and based on these categories, they designed appropriate network architectures for efficient disaster management [12].

III - METHODOLOGY

In this study, the concept of Deep Learning along with Open CV is being employed to ensure early detection of forest fires from a live video feed. The video

feed is being obtained from the First Person View (FPV) camera installed on the UAV. The image frames obtained from both the drone footage are being analysed using a CNN model that has been trained and tested, to detect forest fires in real-time.

The CNN model consists of multiple layers of convolution and pooling layers, followed by fully connected layers. The convolution layers extract features from the images, and the pooling layers reduce the dimensionality of the feature maps. The fully connected layers classify the input into fire or non-fire. The Deep Learning (DL) architecture comprises pre-trained model weights which are obtained from LeNet, AlexNet, VGG-16, VGG-19, Google’s Inception model [10][7]. These models aid in faster training of the huge image dataset. However, in this paper a customized CNN model has been developed without using any of the pre-trained models.

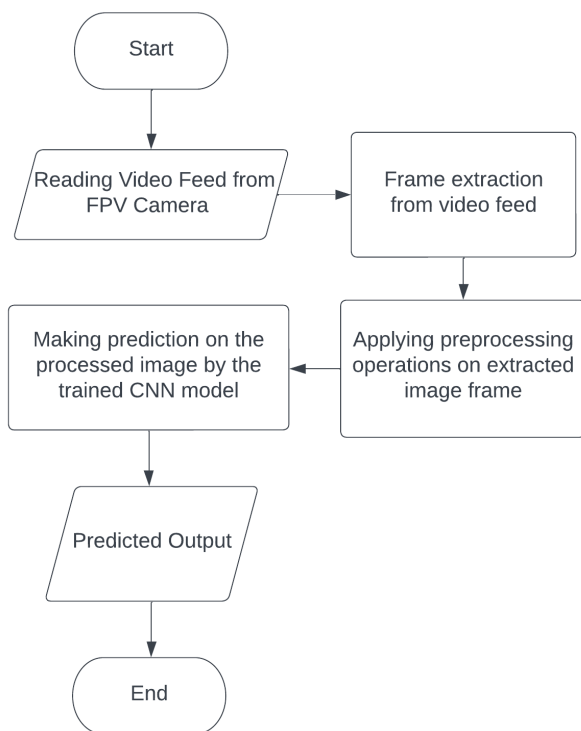


Fig. 3- Methodology flow diagram

The use of various libraries has been made, to assist with the image pre-processing procedures and, to implement the CNN model on the forest fire dataset.

The surveillance drone is to survey the forest areas from an aerial perspective and detect forest fires or any potential threat that may cause fire, like campfires, etc., and alert the central control unit or authorities. The main processing of the drone happens at the Microprocessor and the microcontroller.

A. Tuning Hyper parameters in a convolution neural network (CNN) model using Keras Auto Tuner

Hyper parameters are parameters that are not learned during the training process of a machine learning model but need to be specified beforehand, such as the learning rate, batch size, and number of epochs, regularization strength, and architecture of the model. In CNN, the hyper parameters tuning process is crucial to achieve the best performance of the model. It is important to tune the hyper parameters on a separate validation set to prevent over fitting to the training data. The parameters and the number of epochs for the implemented CNN in this paper has been determined using Keras Auto Tuner [8]. Keras Auto Tuner is a library that is used to automate the hyper parameter tuning process for deep learning models. It is built using the Keras API [5]. A search space is defined for the hyper parameters; which specifies the range of values that each hyper parameter can take. Keras Auto Tuner samples hyper parameters from the search space (specifies the range of values that each hyper parameter can take) and trains the model using each sampled set of hyper parameters. It evaluates the performance of the model on a validation dataset and uses this information to update the probabilistic model of the hyper parameter space. This process is repeated for a specified number of iterations or until a stopping criterion is met, and the best set of hyper parameters found during the search is returned [6].

```

    Model: "sequential_2"
    Layer (type)                Output Shape                Param #
    -----
    conv2d_6 (Conv2D)           (None, 98, 98, 160)        4480
    activation_13 (Activation)   (None, 98, 98, 160)        0
    max_pooling2d_4 (MaxPooling  (None, 49, 49, 160)        0
    2D)
    conv2d_7 (Conv2D)           (None, 47, 47, 96)         138336
    activation_14 (Activation)   (None, 47, 47, 96)         0
    conv2d_8 (Conv2D)           (None, 45, 45, 160)        138400
    activation_15 (Activation)   (None, 45, 45, 160)        0
    max_pooling2d_5 (MaxPooling  (None, 22, 22, 160)        0
    2D)
    flatten_2 (Flatten)         (None, 77440)              0
    dense_7 (Dense)              (None, 512)                 39649792
    activation_16 (Activation)   (None, 512)                 0
    dense_8 (Dense)              (None, 512)                 262656
    activation_17 (Activation)   (None, 512)                 0
    dense_9 (Dense)              (None, 1)                   513
    activation_18 (Activation)   (None, 1)                   0
    =====
    Total params: 40,194,177
    Trainable params: 40,194,177
    Non-trainable params: 0
    
```

Fig. 4- Summary of the CNN model implemented

B. Pre-processing of the images

Fire detection using image processing or computer vision techniques involves several pre-processing steps to improve the accuracy of the detection system. Pre-processing steps can help improve the accuracy and speed of the fire detection system by enhancing the features of the fire and removing noise in the image. However, the specific pre-processing steps may vary depending on the specific fire detection algorithm or application.

In this implementation, the python script has been used to extract the images in JPG (Joint Photographic Experts Group) format from the dataset. The extracted images are then transformed into three dimensional matrices, with the first, second and third dimensions i.e., x, y and c component containing red (x, y, c), green (x, y, c+1) and blue (x, y, c+2) pixel values separately. The sections in the image showing fire are then enhanced by comparing the pixel values of their respective RGB (Red, Green and Blue) components.

The human visual system is most sensitive to changes in the green component of an image, followed by the red component, and then the blue component. Based on this, the approach implemented here for feature extraction is to consider pixels where the red component is greater than the green component, and the green component is greater than the blue component [9]. These pixels are likely to represent areas of the image with high contrast and sharp edges i.e., the region containing fire. Also, the red and green component pixel value should be greater than a particular threshold and the blue component value should be lesser than a particular threshold. This threshold value has been determined through multiple trials and tests.

$$image [x, y, c] > image [x, y, c+1] \quad (1)$$

$$image [x, y, c+1] > image [x, y, c+2] \quad (2)$$

$$image [x, y, c] > 180 \quad (3)$$

$$image [x, y, c+1] > 150 \quad (4)$$

$$image [x, y, c+2] < 100 \quad (5)$$



Fig. 5- Forest fire image before pre-processing [4].



Fig. 6- Forest fire image after pre-processing [4].

The dimensions of the images have also been reduced to 100x100 to reduce the size of the dataset. Resizing the image to a smaller size can help reduce the computational load and speed up the detection process. By performing pre-processing on the images, feature extraction using the CNN becomes easier by generating better feature maps.

C. Surveillance drone flight controlling mechanism

The Microcontroller in combination with the MPU6050 and the BMP388 sensors performs as the flight controller of the drone.

The MPU6050 (Magnetic Pickup Unit 6050) IMU (Inertial Measurement Unit) is a 6-axis gyroscope + accelerometer integrated on a single chip that provides angles in 3 axis- Yaw, Pitch, and Roll. The gyroscope measures rotational velocity or rate of change of the angular position over time, along the X, Y, and Z axis. It uses MEMS (Micro-Electro Mechanical systems) technology and the Carioles Effect for measuring. On the other hand, the MPU6050 accelerometer measures gravitational acceleration along the 3 axes, and by using

trigonometry math the angle at which the sensor is positioned can be calculated. The BMP388 (Barometric Pressure Sensor 388) is a barometric pressure sensor that is used for accurate altitude tracking of the drone.

The flight controller, with the help of PID control loops, sends the signals to the Electronic Speed Controllers (ESCs) that drive the brushless DC motors. There is also a Global Positioning System (GPS) module interfaced with the microprocessor to provide the real-time positional coordinates of the drone. Moreover, an nRF Radio Communication module is used, which acts as a trans-receiver between the drone and the central processing server. Further, a camera is mounted on the drone that transmits the video feed to the control station via an AV transmitter.

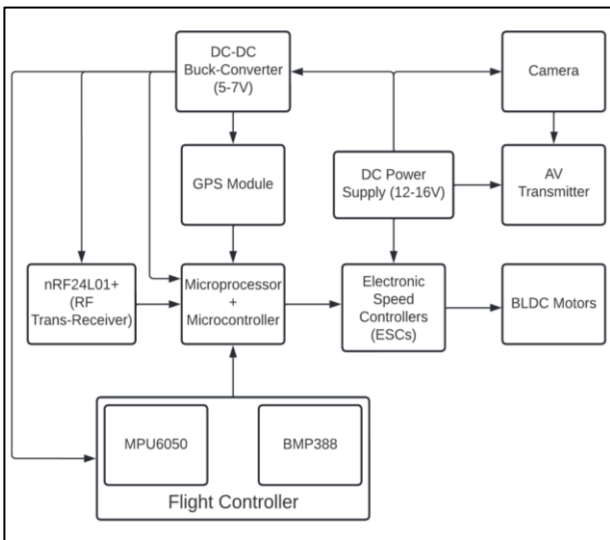


Fig. 7- Block Diagram of the Surveillance drone

The data collected from the MPU6050 and the BMP388 sensors are collected by a microcontroller, and is processed through multiple PID control loops. Based on the errors between the angular setpoints and the current orientation, the PID loops generate a corrective variable for each angular variable - yaw, pitch, and roll. Using these corrective factors, the speed of each of the motors is calculated according to the equations mentioned below.

$$FL = \lambda_o - \theta_o^P + \theta_o^R - \theta_o^Y \quad (6)$$

$$FR = \lambda_o - \theta_o^P - \theta_o^R + \theta_o^Y \quad (7)$$

$$BL = \lambda_o + \theta_o^P + \theta_o^R + \theta_o^Y \quad (8)$$

$$BR = \lambda_o + \theta_o^P - \theta_o^R - \theta_o^Y \quad (9)$$

Where, FL (Front-Left motor), FR(Front-Right motor), BL(Back-Left motor), BR(Back-Right motor) represent the speed of the individual motors and $\lambda_o, \theta_o^P, \theta_o^R, \theta_o^Y$ represent the output value for altitude, pitch, roll and yaw action [13].

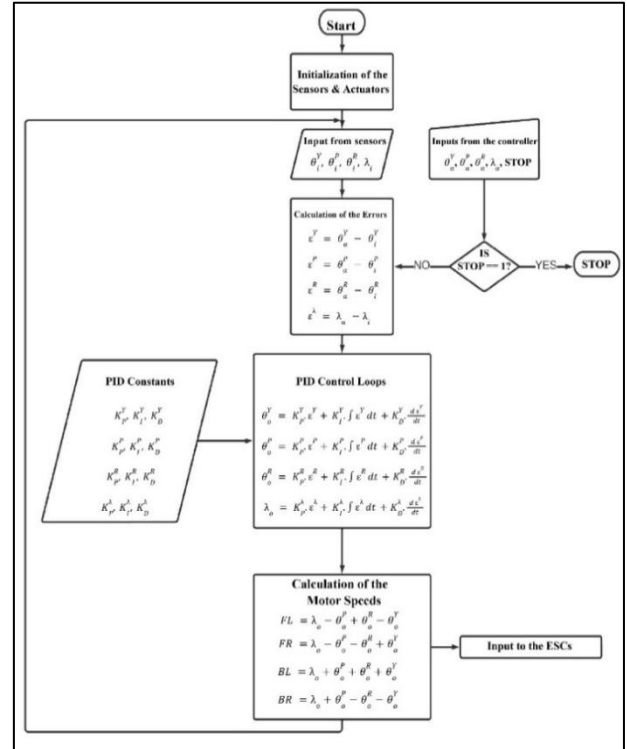


Fig. 8- Flowchart for the flight controller algorithm

Once the power supply is switched ON, the sensors and the ESCs are initialized. In the initialization stage, the offsets for the MPU6050 sensor are calculated and applied in order to provide accurate gyroscopic data. Also, the ESCs are given the minimum pulse for initialization, making the ESCs enter into operational mode. Further, the sensors continuously provide the angular as well as altitude orientation of the drone to the microcontroller. The microcontroller receives the input signals in the form of setpoints from the radio receiver. The microcontroller calculates the error or difference between the setpoints and the current orientational parameters and feeds these errors to the PID control loops. The PID controllers calculate the corrective output variables for each angular and altitude parameter using the traditional PID equation.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (10)$$

Where, $K_p, K_i,$ and K_d (all non-negative) denote the coefficients for the proportional, integral, and derivative terms respectively.

Using these corrective factors, the output speed for each motor, in terms of pulse signals, is calculated according to the equations stated above. This process keeps on repeating in a loop until the STOP command is received through the radio receiver.

The tuning of the PID constants for the drone is being carried out by mounting the drone on a test rig that enables motion in Yaw, Pitch, Roll, and vertical orientations. Employing the traditional manual tuning method that involves starting by setting all the parameters to 0, and then progressively raising the value of the proportional constants until sustained oscillations are obtained. Once the proportional constant is tuned nearly to the desired value, the drone oscillates at a constant rate around the setpoints. Further, the derivative constants reduce the rate of change of the errors, resulting in a faster response to the change in the orientation of the drone. This in turn damps the oscillations, hence reducing the steady-state oscillations and stabilizing the drone. As the derivative term is susceptible to noise in the input signals, even a small noise can amplify the output of the derivative controllers. Hence, to reduce the overtime steady-state error, integral constants need to be introduced [14].

Table 2- Legend table for all the symbols used in the flight controller algorithm flowchart

Legends:		
θ = Angle ε = Error λ = Altitude	Superscripts: Y = Yaw P = Pitch R = Roll λ = Altitude	Subscripts: i = Input α = Setpoint o = Output
PID Constants: K_P = Proportional Constant K_I = Integral Constant K_D = Derivative Constant	Motor speeds: FL = Front-Left Motor FR = Front-Right Motor BL = Back-Left (Rear-Left) Motor BR = Back-Right (Rear-Right) Motor	Control Signal: STOP = For exiting the loop and stopping the drone.

IV- RESULTS & DISCUSSIONS

The CNN model is designed to analyse drone footages and detect forest fires in real-time, which can help to improve early warning and response systems and reduce the impact of wildfires.

The model performance was evaluated on a dataset of forest fire images obtained from Kaggle [4]. It attained up to 91.18 percent accuracy with a loss of 0.42 after training with an epoch size of 50.

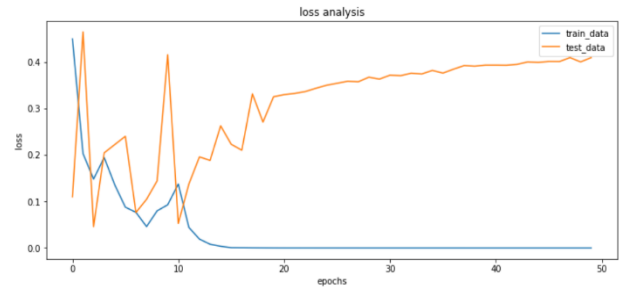


Fig. 9- Loss analysis for the CNN model

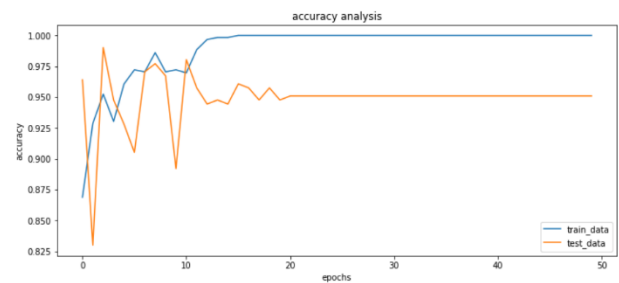


Fig. 10- Accuracy analysis for the CNN model

The model could be optimized further to give minimum number of false positives as well as false negatives.

Table 3- Results from the confusion matrix of the CNN model

False positives	False negatives	True positives	True negatives
8	25	165	176

The surveillance drone was able to operate wirelessly with manual controls. The drone transmitted the video feeds to the server (laptop) via AV transmitter. The laptop processed these video feeds and predicted the presence of fire using CNN. If fire was detected, the current GPS coordinates of the drone will be recorded on the server, and the authorities are alerted about the situation.

This method has significant potential for real-world applications, especially in remote areas where traditional ground-based monitoring systems are not feasible. CNN-based approach can be applied to a variety of satellite imagery datasets and can be integrated with existing fire monitoring systems to improve their accuracy and efficiency.

V-CONCLUSION

In this paper, we present an efficient solution to the problem of forest fire detection in the area of disaster management. We provide an automated ML-based system which incorporates surveillance using UAVs. The novelty of our work lies in the simple flight-controller algorithm, integrated with the coordinated communication algorithm with the aerial and ground vehicles. We give guarantees of latency and accuracy. The existing ML and CV methods like CNN and YOLO fit in very well in the overall system for giving early warning of forest fires.

At present the work is in the state of proof-of-concept and in near future, we intend to build a real-life system by taking actual field tests and improving the accuracy of the ML sub-part. The battery-life and payload improvements also can be brought in by some state-of-the-art Drone usage.

Despite the current challenges and limitations in technology, the application has a lot of potential to revolutionize safety and disaster management scenarios.

REFERENCES

- [1] N. S. R. k. C. R. G. N. H. R. Shamama Afreen, "Forest policies and programs affecting vulnerability and adaptation to climate change," *Springer Nature*, 2010.
- [2] Q. L. C. Q. H. G. Yihong Zhang, "Forest Fire Smoke Detection Method Based on MoAm-YOLOv4 Algorithm," *Journal of Computer and Communications*, 2022.
- [3] S. K. G. A. G. Anuraag Biswasa, "Early Fire Detection and Alert System using Modified Inception-v3," *Procedia Computer Science*, vol. 218, pp. 2243-2252, 2023.
- [4] B. Dincer, "Wildfire detection image data," *Kaggle*, 18-May-2021. [Online]. Available: <https://www.kaggle.com/datasets/brsdincer/wildfire-detection-image-data>. [Accessed: 01-Mar-2023].
- [5] R. Dwivedi, "How To Use Keras Tuner for Hyperparameter Tuning of Deep Learning Models," *Analytics India Magazine*, 20-Aug-2020.
- [6] C. Conol, "Hyperparameter Tuning with Keras Tuner," *Towards Data Science*, 01-May-2020.
- [7] W. L. Y. J. P. S. S. R. D. A. D. E. V. V. A. R. Christian Szegegy, "Going deeper with convolutions," *arXiv*, 2014.
- [8] S. Joshi, J. A. Owens, S. Shah and T. Munasinghe, "Analysis of Preprocessing Techniques, Keras Tuner, and Transfer Learning on Cloud Street image data," *2021 IEEE International Conference on Big Data (Big Data)*, Orlando, FL, USA, 2021, pp. 4165-4168, doi: 10.1109/BigData52589.2021.9671878.
- [9] N. Petrellis, "A Review of Image Processing Techniques Common in Human and Plant Disease Diagnosis," *Symmetry*, vol. 10, no. 7, p. 270, Jul. 2018, doi: 10.3390/sym10070270.
- [10] M. Swapna, Y. K. Sharma, and B. M. G. Prasadh, "CNN Architectures: Alex Net, Le Net, VGG, Google Net, Res Net," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 8, no. 6, Mar. 2020.
- [11] S. Sudhakar, V. Vijayakumar, C. S. Kumar, V. Priya, L. Ravi, and V. Subramaniaswamy, "Unmanned Aerial Vehicle (UAV) based forest fire detection and monitoring for reducing false alarms in forest-fires," *Computer Communications*, 04-Oct-2019. [Online]. Available: <https://www.sciencedirect.com/science/article/abs/pii/S0140366419308655>. [Accessed: 20-Jun-2022].
- [12] "Help from the sky: Leveraging uavs for Disaster Management - ResearchGate." [Online]. Available: https://www.researchgate.net/publication/312254141_Help_from_the_Sky_Leveraging_UAVs_for_Disaster_Management. [Accessed: 15-Apr-2022].
- [13] "The four basic flight modes of drones. A vertical lift, B rolling ..." [Online]. Available: https://www.researchgate.net/figure/The-four-basic-flight-modes-of-drones-a-Vertical-lift-b-rolling-rotation-c-pitching_fig5_315112663. [Accessed: 18-Jun-2022].
- [14] Pachputesharad@gmail.com, "Working Principle and components of drone · CFD Flow Engineering," *CFD Flow Engineering*, 15-Oct-2022. [Online]. Available: <https://cfdfloengineering.com/working-principle-and-components-of-drone/>. [Accessed: 14-Jun-2022].