CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Forest fire are a matter of concern because they cause extensive damage to environment, property and human life. Hence, it is crucial to detect the forest fire at an earlier stage. This can help in saving flora and fauna of the region along with the resources. Also, it may help to control the spread of fire at initial phase. The task of monitoring the forests is difficult because of the vast territory and dense forest.

The wide ranging adverse ecological, economic and social impacts of forest fire including forest degradation are:

- Loss of valuable wood resources
- Deterioration of catchment areas
- Loss of biodiversity and extermination of flora and fauna
- Loss of wildlife habitation and exhaustion of wildlife
- Global warming

The forest fire has become a threat to not only to the forest wealth but also flora and fauna and ecology of the environment of the region. The main cause of forest fire can be categorized under natural and man-made classes. High atmospheric temperature, lightening and dryness (low humidity) offer positive environment for a fire to start which are the natural causes for forest fire. The fire is also caused by Man-made sources like naked flame, cigarette, electric spark, etc.

1.2 FOREST FIRE

It is very essential to study the forest fire to ratiocinate its background and to provide the justification for undertaking of study. The forest fires are the most uncontrollable event causes critical disturbance to complete ecosystem that needs to be tackled implementing WSNs technology. The following reason is the driving force behind for the undertaking of this research in order to prevent forest from fires. Due to the presence of various other problems, human civilization affects biological resources massively and ultimately leads to the degradation of biological diversity. There is a strong requirement of planning forest management with efficient tools for the preservation of biological diversity. India is a country having rich biodiversity of flora and fauna. The degradation of forests due to forest fires and other activities affects wildlife and poses threat to habitat which has been a matter of concern. An act (Wildlife Protection Act, 1972) for protecting wildlife, animals, birds, plants and things that are connected to wildlife has constituted in order to ensure the environmental and biological safety of the country.

The establishment and supervision of protected areas considering the optimal use of resources, conservation and rehabilitation of ecosystem are addressed in article 8 of CBD (Convention on biological diversity). The protected areas of a country are very valuable and of great significance for the preservation of natural and cultural resources. The good conservation of protected areas provides various opportunities and benefits in terms of ecological, educational, economic and social values. However, these protected area in a country are exposed to various threats. The most common hazard that happens in a forest is wildfire that completely disturbs and destroys ecosystem and wildlife. The occurrence of wildfire across the globe in increasing every year that causes a great deal of threat to entire biodiversity.

1.2.1 Terminology of Forest Fire

Ever since the science starts to begin the terms fire risk and fire hazard are related to forest fire management. There found various different expressions and definitions for these two terms. The proper understanding of fire management terms such as fire hazard, danger, risk, vulnerability, etc. is important as wrong interpretation may lead to misunderstanding from their real meanings. Fire risk is defined as the combination of fire likelihood chances and the consequences of identified fire hazard. It is used for measuring the loss and harm from a fire activity. Fire risk determines as the probability dangerous consequences of fire and expected loss for damage assessment in terms of loss of lives and natural property. The prediction of fire risk is expressed by the equation shown below.

$$Fire\ Risk = Fire\ Hazard \times Fire\ Vulnerability$$

Where, fire hazard is the characteristic of fire condition and fire vulnerability is the amount of loss. Fire hazard is a characteristic of fire condition that causes damage to environment, property, wildlife and people. The definition of fire hazard states that it's a phenomena or manmade activity has the potential of physically damaging the environment which results in loss of wildlife, loss of natural property, and leads to the degradation of environment. Fire vulnerability is the amount of loss takes place to a specified portion at risk and fire vulnerability is measured as 0 for zero damage and 1 for overall damage. Fire vulnerability is a combination of conditions and processes which are designed considering various environmental, physical, economic factors for evaluating the impacts of hazard.

1.2.2 Indian Scenario of Forest Fire

The forest cover of India is around 21.6% out of total area cover of India which is approximate 3.3 million Km2. The geometric location of India makes it a country with mega biodiversity. India has different climate regions which includes Himalayan region at its northeast, tropical zone at south and desserts in northwest part of the country. The vegetation in Indian forests varies form evergreen forests of south west region and alpine forests in north region. There exists many semi-green, tropical hill and pine forests in between this two-extreme vegetation of forests. Due to the involvement of humans and their lifestyles the forests in India declining very fast. The vulnerability of fire incidents in Indian forests differs from one place to another based on the kind of climate and vegetation. On the basis of forest records, it is observed that more than 50% of forests in India are vulnerable to fire and Rs. 450 crores are Indian annual loss due to forest fires. Over 10,500 incidents of forest fire have been reported in the year of 2016 which are five times more than the reported events of previous year.

1.2.3 Forest Fire Impacts

Most of the reported fire incidents in India caused by human interventions. These fires adversely affect Indian forests in many perspectives such as social, economic and ecological impacts. The adverse impact of forest fires in India includes loss of vegetation such as fuel wood and timber, loss of natural habitat such as microorganisms, wildlife and biodiversity. The average estimated loss due to forest fires is very high which strong demands for such a system that can reduce the cause or stop it. The fire tolerance of trees in thick forests ensures the long-term fire exposure in forest. It is believed that due to human interventions in forest for grazing and triggering intentional fire for clearing field’s further leads to the degradation of forest areas. This intervention of human in forests for making goods leads to the conversion of thick forests to dry grasslands and scrublands and becomes highly prone to fires. It is also observed that with the increase in forest utility by humans, the noticed wildfire events also increased. The wildfires are significantly impacting the forest areas. The flames of fire is the reason behind killing of major vegetation. The soil for a short period of time becomes more like a concrete surface after wildfire incidents. The healthy trees and roots usually suck up the moisture when it rains but after wildfire the water has nowhere to go. Overtime the landscape could change entirely, with trees replaced by shrubs and grasses. Wildfires are usually a necessary part of the regeneration cycle but as wildfires become so frequent, it poses a great deal of threat to complete ecosystem. Forests plays a huge critical role in capturing carbon and taking in more greenhouse gases. The more forest fires are burning more carbon gets released and more accelerating of climate changes happening.

1.2.4 Causes Responsible

Forest fire is an uncontrollable fire that happens in nature and causes threat to forest wealth and entire regime. The two major causes responsible for forest fires are environmental and human related forest fires. The natural fire happens due to natural calamities whereas human related fire occurs due to the intervention of human intentionally or unintentionally. Forest fire is one of the major reasons behind forest degradation. At present, there are many agencies working for the conservation of forest wealth. Their efforts can only be successful if the factors causing of deforestation are not taken into account.

1.3 PURPOSE OF THE PROJECT

For a successful fire management program early detection plays an important role. Fire attack at a very initial stage depends on time, delay, various fuels, weather and the size of fire. Fire detection in its very early stage increases the probability that it can be controlled before it converts to a great damage or an uncontrollable event and early detection also reduces suppression costs and negative environmental impacts. Fire monitoring is the process of characterizing and mapping the parameters of a fire which includes the information of location, fire perimeter, active fronts, size proximity and inhabited areas that helps to know the present status and changes of an active fire. Once the decision has been made to begin the suppression, the importance is then given to fire monitoring which is very important and critical for suppression planning and public warnings. Fire detection systems that have been developed are used for both actions that is detection and monitoring phases.

1.4 NEED OF FOREST FIRE DETECTION

Forests are one of the most important resources and necessary part of our ecosystem. Humans for their survival are completely dependent on forests from the fresh air we breathe to the use of natural products that we rely on. The forest fire incident happens because of natural causes and manmade activities. Every healthy forest contains some dead plants, trees that are prone to fire easily. Forest fires threatens the wildlife directly and the release of poisonous gases during fire event can affect all lives. Every individual is responsible for protecting the environment from forest fire disaster. Every year from fire million hectares of land are destroyed causing some serious damages to the natural environment. Fire detection system is very important for the detection of fire to stop the cause or to reduce the effect of fires. The continuous monitoring of a fire prone area and predicting fire at its initial stage can significantly reduce the risk of fire as well as firefighting cost.

1.5 OBJECTIVES OF THE PROJECT

Our primary objective is to detect forest fires as early as possible. Early detection allows for swift response, potentially minimizing the size and impact of the fire. Once a fire is detected, the goal is to respond quickly and effectively. This includes deploying firefighting resources to contain and extinguish the fire before it spreads further. Detecting forest fires promptly helps to protect human life by giving people in the affected area more time to evacuate safely. Another objective is to protect property, including homes, infrastructure, and natural resources, from the destructive effects of forest fires. Forest fires can contribute to air and water pollution, as well as soil erosion. Detecting fires early can help mitigate these environmental impacts. Continuously monitoring and detecting forest fires, authorities can gather valuable data to improve fire management strategies and prevent future fires. Detecting forest fires also serves to raise public awareness about the dangers of wildfires and the importance of fire prevention and safety measures. Advances in forest fire detection technologies and techniques are ongoing. One objective is to continue researching and developing innovative methods for detecting and monitoring forest fires more effectively.

1.6 SCOPE

The scope of a forest fire detection project can vary depending on factors such as the resources available, the specific objective, and the geographic area being covered. Define the geographic area or areas where the forest fire detection project will be implemented. This could be a specific region, a National park, or even an entire country. Conduct a thorough assessment of the risk of Forest Fires in the study area. This may involve analyzing historical data on past fires, weather patterns, vegetation types, human activities, and other relevant factors. Evaluate and select appropriate technologies for forest fire detection. This could include satellite imagery, remote sensing systems, ground-based sensors, drones, and/or artificial intelligence algorithms. Design a comprehensive system for forest fire detection and monitoring. This includes determining the placement of detection devices, establishing communication networks, and developing protocols for data collection and analysis. Develop response plans and protocols for how to react to detected forest fires. This includes coordinating firefighting efforts, evacuations, and other emergency measures. Ensure the long-term sustainability of the forest fire detection project by establishing maintenance schedules, securing funding, and building partnerships with relevant organizations and agencies. Raise public awareness about the importance of forest fire prevention and safety through educational campaigns, outreach events, and community engagement initiatives.

CHAPTER 2

LITERATURE SURVEY

2.1 TITLE: Wildfire smoke detection based on co-occurrence matrix and dynamic features

AUTHORS: Hoai Luu-Duc,Dung Trung Vo; 2021

The paper presents a new approach to detect wildfire smoke in forest by using camera surveillance. The basic idea of the proposed method is that smoke is grayish and nonrigid object, it normally creates chaos information in image. The new approach includes three steps, the pre-processing step to reduce noise as well as to divide the image into small blocks, the determining candidate smoke objects step using smoke color detection and slow motion detection, and the analysing smoke objects step using Co-occurrence matrix to extract data from candidate smoke objects.

ADVANTAGE:

- The experiment results on wildfire smoke videos show that Co-occurrence matrix can be used to extract smoke features and it can be combine with other extracting features method to improve the accuracy of the wildfire smoke detection method.

DISADVANTAGE:

- Very low accuracy of the wildfire smoke detection

2.2 TITLE: Vision based wildfire and natural risk observers

AUTHORS: Darko Stipanicev,Ljiljana Seric,Maja Braovic; 2020

Wildfires are natural risk phenomena that cause significant economic and environmental damage. In wildfire fighting strategy it is important to detect the wildfire in its initial stage and to apply, as soon as possible, the most appropriate firefighting action. In both cases wildfire monitoring and surveillance systems are of great importance, so in the last decade the interest for various wildfire monitoring and surveillance systems has increased, both on the research and the implementation level. This paper describes one such system named iForestFire. It is an example of advanced terrestrial vision-based wildfire monitoring and surveillance system, today widely used in various Croatian National and Nature Parks and regions, but it is also a system in constant development and improvement, both on theoretical and practical level. This paper describes its last improvements in video detection part that are based on notation of observer, cogent confabulation theory and mechanism of thought.

ADVANTAGE:

- Inclusion of cogent confabulation theory allows us to expend the use of existing wildfire observers to more general natural risk observers.

DISADVANTAGE:

- Algorithm takes more time to get result

2.3 TITLE: Quantum-Compatible Variational Segmentation for Image-to-Image Wildfire Detection Using Satellite Data

AUTHORS: Ata Akbari Asanjan, Milad Memarzadeh; 2022

Wildfire occurrences have been increasing for the past decade, leaving devastating traces across the world. In the recent efforts, remote sensing and airborne missions have been utilized to better understand and manage wildfires. This has resulted in an exponential increase in volume of remote sensing data, which has pushed the need for intelligent automation of data extraction for wildfire studies. Machine learning offers accu-rate automation in detecting such natural anomalies and en-able decision-makers to take actions in a timely manner. Re-cent advances in machine learning algorithms, namely probabilistic generative methods, allow researchers and decision- makers to step beyond detection and study “what-if’ scenarios for wildfire occurrences. Additionally, they offer better imitations to the stochastic behavior of nature, and wildfire events. However, optimizing the performance of these probabilistic generative models is a computationally expensive pro-cess, specially using digital computers. On the other hand, quantum computers have recently shown a promise to reduce computationally costly training of such models and provide performance improvements. There is a body of research investigating the potential for improved machine learning methods in which key operations are performed on a quantum computer. In this study, we propose a probabilistic image-to- image segmentation approach combining a very well-known segmentation method, U-NET, with a Conditional Variational Auto-Encoder (CVAE) to not only detect wildfires but also describe the stochasticity of the phenomenon and be capable of running “what-if’ scenarios.

ADVANTAGE:

- The proposed model is compatible with training on quantum computers, which results in a quantum-assisted image-to-image segmentation approach and can be used to benchmark the potential benefit of quantum computing over the classical one.

DISADVANTAGE:

- Quantum computing over the classical one and scalability of our method

2.4 TITLE: Low-Power Distributed Sensor Network for Wildfire Detection

AUTHORS: Triston Blalack, Dakota Ellis,Marcus Long; 2021

Wildfires kill and injure people, destroy residences, pollute the air, and cause economic loss. In this paper, a low-power Internet of Things (IoT)-based sensor network is developed, which automatically detects fires in forests and sends the location to a central monitoring station with smartphone notifications in a real-time setting. This action helps in the early detection of a fire and firefighters can be notified immediately—thus the spread of the fire and the harm caused by it can be reduced. The proposed system detects fires from the presence of smoke and a sudden increase in temperature. The system also logs the temperature, humidity, carbon dioxide, rain, light, and wind speed in different areas of the forest. The sensor nodes transmit the data to a hub using a long-range wireless transmitter and the hub then sends the data to the central monitoring station using the cellular Internet. The sensor nodes and hub are designed with ultra-low-power hardware and software architecture, consuming current of only 0.37 and 1.4 mA, respectively, so that they can be powered by solar panels throughout the year.

ADVANTAGE:

- The central server and smartphone app contain maps, and the wildfire locations are marked in the case of a fire.
- In the present study, a prototype of the proposed system is successfully developed and tested.

DISADVANTAGE:

- Prediction accuracy rate is low

2.5 TITLE: Early Wildfire Detection Based on Temporal, Spatial and Spectral Information Fusion

AUTHORS: Qiang Zhang,Jian Zhu; 2023

In this work, we propose a novel framework for near-real-time and early-stage wildfire detection using Himawari-8 satellite 10-minute data. Dispense with cloud detection and setting too many manual thresholds. The proposed framework jointly combines spatial, temporal and spectral information for wildfire detection. Compared with JAXA’s wildfire products, the proposed framework can more accurately detect the wildfire points. Especially for the early-stage wildfire detection.

ADVANTAGE:

- Improve accuracy

DISADVANTAGE:

- The simulation outcome delay

2.6 TITLE: Early Detection of Wildfires in the European Area with Nano-Satellite Constellations

AUTHORS: Bilge Memis,Bengisu Kaplan;2020

According to the report published by the Turkish Ministry of Agriculture and Rural Affairs, areas burned by forest and wildland fires reached a record high in 2021. Likewise, the area burned by forest and wildland fires is increased compared to the last ten years' average in Europe in 2022. The total area burned is directly related to the response time to fires. Therefore, it is essential to decrease the fire detection times to decrease the burned land. In the scope of this paper, a satellite constellation-based wildfire detection system is examined. Different constellations with different orbit parameters are considered. Results are evaluated from the point of cost and efficiency.

ADVANTAGE:

- Revisit times for the specified areas are calculated to prove the short fire detection times, better than 10 minutes are possible with the proposed constellation.

DISADVANTAGE:

- Time consuming method

2.7 TITLE: Early Detection of Wildfires with GOES-R Time-Series and Deep GRU Network

AUTHORS: B D Parameshachari, G M Siddesh,V. Sridhar, 2022

Early detection of wildfires has been limited using the sun-synchronous orbit satellites due to their low temporal resolution and wildfires' fast spread in the early stage. NOAA's geostationary weather satellites GOES-R Advanced Baseline Imager (ABI) can acquire images every 15 min at 2 km spatial resolution, and have been used for early fire detection. However, advanced processing algorithms are needed to provide timely and reliable detection of wildfires. In this research, a deep learning framework, based on Gated Recurrent Units (GRU), is proposed to detect wildfires at early stage using GOES-R dense time series data. GRU model maintains good performance on temporal modelling while keep a simple architecture, makes it suitable to efficiently process time-series data. 36 different wildfires in North and South America under the coverage of GOES-R satellites are selected to assess the effectiveness of the GRU method. The detection times based on GOES-R are compared with VIIRS active fire products at 375 m resolution in NASA's Fire Information for Resource Management System (FIRMS). The results show that GRU-based GOES-R detections of the wildfires are earlier than that of the VIIRS active fire products in most of the study areas.

ADVANTAGE:

- The results from proposed method offer more precise location on the active fire at early stage than GOES-R Active Fire Product in mid-latitude and low-latitude regions.

DISADVANTAGE:

- Time provide sufficiently high accuracy on the burned areas.

2.8 TITLE: FireSpot: A Database for Smoke Detection in Early-stage Wildfires

AUTHORS: Chenchen Li, Yan Li, Yubin Ba;2023

Forests are an essential natural resource to humankind, providing a myriad of direct and indirect benefits. Natural disasters like forest fires have a major impact on global warming and the continued existence of life on Earth. Automatic identification of forest fires is thus an important field to research in order to minimize disasters. Early fire detection can also help decision-makers plan mitigation methods and extinguishing tactics. This research looks at fire/smoke detection from images using AI-based computer vision techniques. Convolutional Neural Networks (CNN) are a type of Artificial Intelligence (AI) approach that have been shown to outperform state-of-the-art methods in image classification and other computer vision tasks, but their training time can be prohibitive. Further, a pretrained CNN may underperform when there is no sufficient dataset available. To address this issue, transfer learning is exercised on pre-trained models. However, the models may lose their classification abilities on the original datasets when transfer learning is applied.

ADVANTAGE

- Use learning without forgetting (LwF), which trains the network with a new task but keeps the network's preexisting abilities intact.

DISADVANTAGE

- High cost method

2.9 TITLE: Wildfire Detection and Avoidance of false Alarm Using Densenet

AUTHORS: Sridhar P, Rexna Devi N;2022

The environmental crisis leaves human livelihood at stake. One such catastrophic situation humans tend to face is wildfire, and the damages it creates are uncertain. Fire-like objects and small-size fire objects have a high false alarm rate. Wildfire is one of the most common hazards and natural threats to biodiversity, disturbing the ecology and environment of a region. Natural causes such as lightning and man-made causes are significant sources of forest fires. If the wildfire goes unnoticed, it cannot be controlled quickly, and hence the impact of these disastrous effects can result in massive damage to human properties and areas of vegetation. On average, 62,963 wildfires occur worldwide annually and affect millions of hectares. And such impacts could be avoided if there were systems available for monitoring and early detection of the wildfire. Flame detectors, smoke detectors, and temperature detectors are examples of early fire detection technologies that are inadequate in the event of a widespread wildfire. Sensor-based and image-processing-based technology is two types of existent technology. Hence we initiate architecture with DenseNet for wildfire detection and avoidance of false alarms, using a mixed dataset of fire-like objects and non-fire objects which is definitive for wildfire detection excluding false alarms

ADVANTAGE

- With the help of real-time surveillance cameras, constant and accurate monitoring under deep learning-based automatic wildfire detection is possible.

DISADVANTAGE

- DenseNet for wildfire detection and avoidance of false alarms, using a mixed dataset of fire-like objects and non-fire objects which is definitive for wildfire detection excluding false alarms

2.10 TITLE: A Multi-temporal Anomaly Analysis Wildfire Detection Method for Transmission Lines

AUTHORS: Limei Ma,Yijun Gao, Chen Zhao; 2021

Fengyun 4A (FY-4A), the first of China's second-generation geostationary orbiting weather satellites carries a novel optical instrument named Advanced Geosynchronous Radiation Imager (AGRI). An active wildfire detection method based on the AGRI data was developed and tested in this paper. Most of the existing wildfire detection methods were developed on the spectral features of pixels while the temporal characteristics of observed values were ignored. Wildfire pixels can be detected by analyzing characteristics of variation trend value and contextual tests confirmation. To assess the performance of proposed method, the corridor areas within 3 km of electrical transmission lines in China were selected as study areas.

ADVANTAGE

- Wildfire detection results demonstrate the effectiveness of proposed method with higher sensitivity to detect even small fires and a low wildfire omission error rate.

DISADVANTAGE

Higher sensitivity to detect even small fires and a low wildfire omission error rate.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Active Fire (AF) detection and burnt area (BA) segmentation using Remote Sensing techniques have been in focus of many research works since as early as the 1970s with the launch of Landsat-1 in 1972. To better understand Fire detection from space, it is important first to distinguish between two types of products that are the goal of most research works in the field. In recent years, several algorithms that attempt to combine sensors from multiple spectral domains were suggested. These methods have several potential advantages: First, they allow us to exploit the cloud-penetrating capabilities of microwave sensors in combination with the ability of sensors in the visible light and infrared domains to detect thermal anomalies.

Second, combining multiple sensors improves the satellite overpass frequency over a burning area. Finally, an improved temporal and spatial resolution BA and AF products can potentially be achieved by combining images from high-spatial-resolution sensors such as Sentinel-2 with high-temporal-resolution sensors such as Sentinel-3 or sensors on board geostationary satellites. Verhegghen et al. were one of the first to suggest combining time series of SAR C-band images from Sentinel-1 with multispectral imagery from Sentinel-2 to improve BA estimation in cloudy conditions. BAs were detected in Sentinel-2 images using thresholding of spectral indices, and significant changes in the backscattering coefficient of Sentinel-1 images were used as a gap-filling detector. Crowley et al. applied Bayes' theorem approachto combine potential BA detections from Landsat8 OLI, Sentinel-2 MSI, and MODIS instruments to delineate the BA and monitor the progress of the 2017 Elephant Hill fire in California. One of the most recent studies by Lasko combined Sentinel-1 SAR imagery with a monthly averaged MODIS BA product to reduce the burn date uncertainty of the MODIS BA product caused by observations obscured by clouds. Significant decrease in backscatter between two SAR images was used to detect BA-affected pixels in SAR images, which allowed to decrease the burning date uncertainty

3.1.1 Disadvantages of Existing System

- One of the main disadvantages of SAR technology is its lower spatial resolution compared to other types of satellite imaging.
- Spatial resolution refers to the level of detail that can be seen in an image and is typically measured in meters per pixel.

3.2 PROPOSED SYSTEM

Forest Fires are a common occurrence worldwide due to climate change, which results in severe economic losses and ecological destruction. Forest fires can be natural or man-made forest fires and summer forest fires caused by debris and other biomes, as well as human negligence. Even though, wildfires can benefit local vegetation, animals and ecosystems, but they can also cause major damage to property and human life. In recent years, the frequency of forest fire accidents has been continuously increasing. Hence, there has been a rise in interest of implementing systems for automated observation and detection of forest fires, as a means of protecting forests from destruction. Due to the complex background and a number of interference in the original image, the result of the training is not so good. A method is proposed to segment the candidate flame region based on the color feature, and then part of the image is sent to the CNN network for training, which can extract features more specifically and improve the recognition rate of forest fire image effectively. In the training phase, firstly, the binary image of the suspected flame region is segmented, and the result obtained by performing AND operation between the binary image and the original image is used as a training set, and a label is set for each image. A network model is obtained after training the CNN according to the training set. In the testing phase, similarly, the binary image of the suspected flame region is firstly segmented, and the result obtained by performing AND operation with the original image is used as a testing set.

3.2.1 Advantages of Proposed System

- Convolutions are not densely connected; not all input nodes affect all output nodes.
- This gives convolutional layers more flexibility in learning.
- Moreover, the number of weights per layer is a lot smaller, which helps with high-dimensional inputs such as image data.

CHAPTER 4

SYSTEM DESIGN

4.1 SOFTWARE REQUIREMENTS

- Operating system : Windows XP/7
- Coding Language : PYTHON
- Tool : Anaconda Navigator

4.2 HARDWARE CONSTRAINTS

- System : Pentium IV 2.4 GHz
- Hard Disk : 40 GB
- Floppy Drive : 1.44 MB
- Monitor : 15 VGA Colour
- Mouse : Logitech
- Ram : 512 MB

4.3 SOFTWARE DESCRIPTION

Python

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Features of Python

- **Simple**

  Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

- **Easy to Learn**

  As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

- **Free and Open Source**

  Python is an example of a *FLOSS* (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

- **High-level Language**

  When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

- **Portable**

  Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like Kivy to create games for your computer *and* for iPhone, iPad, and Android.

- **Interpreted**

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just *run* the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

- **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In *procedure-oriented* languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In *object-oriented* languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

- **Extensible**

  If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

- **Embeddable**

  You can embed Python within your C/C++ programs to give *scripting* capabilities for your program's users.

- **Extensive Libraries**

  The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the *Batteries Included* philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

Writing Desktop Application

Python has emerged as a powerful and versatile language for developing desktop applications, offering developers an array of frameworks and libraries to streamline the process. One prominent framework is PyQt, which allows developers to create cross-platform desktop applications with a native look and feel. PyQt combines Python's simplicity with the capabilities of the Qt framework, providing a robust toolkit for designing graphical user interfaces (GUIs). With the ability to handle events, manage widgets, and create interactive elements, PyQt facilitates the development of intuitive and visually appealing desktop applications. Additionally, the framework supports a range of features, including multi-threading, networking, and database integration, making it suitable for a broad spectrum of desktop application development needs.

Tkinter

Tkinter, the standard GUI toolkit for Python, facilitates the development of graphical user interfaces in desktop applications. It acts as a bridge between Python and the Tk GUI toolkit, offering a comprehensive set of tools and modules. Tkinter includes various widgets like buttons, labels, and textboxes that allow developers to construct interactive user interfaces. Operating on an event-driven paradigm, Tkinter responds to user actions, enabling the creation of dynamic and responsive applications. Tkinter's extensibility allows developers to create custom widgets and enhance existing ones to meet specific application requirements. The introduction of the `ttk` (themed Tkinter) module further enhances the toolkit's visual appeal, offering themed widgets for a modern appearance. In summary, Tkinter stands out as a powerful and accessible tool for Python developers, enabling the creation of desktop applications with user-friendly graphical interfaces, simplicity, and cross-platform compatibility.

FLASK

Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality. Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality.

1. Flask is a **lightweight** backend framework with minimal dependencies.

2. Flask is **easy to learn** because its simple and intuitive API makes it easy to learn and use for beginners.
3. Flask is a **flexible Framework** because it allows you to customize and extend the framework to suit your needs easily.
4. Flask can be used with **any database** like:- SQL and NoSQL and with **any Frontend Technology** such as React or Angular.
5. Flask is **great for small to medium projects** that do not require the complexity of a large framework.
6. Flask Documentation

Jupyter Notebook

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality. Project Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Jupyter will always be 100% open-source software, free for all to use and released under the liberal terms of the modified BSD license.

Jupyter is developed in the open on GitHub, through the consensus of the Jupyter community. For more information on our governance, please see our governance documentation. All online and in-person interactions and communications directly related to the project are covered by the Jupyter Code of Conduct. This Code of Conduct sets expectations to enable a diverse community of users and contributors to participate in the project with respect and safety.

kernel

A notebook kernel is a “computational engine” that executes the code contained in a Notebook document. The ipython kernel, referenced in this guide, executes python code. Kernels for many other languages exist (official kernels).

Jupyter Notebook is a notebook authoring application, under the Project Jupyter umbrella. Built on the power of the computational notebook format, Jupyter Notebook offers fast, interactive new ways to prototype and explain your code, explore and visualize your data, and share your ideas with others. Notebooks extend the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results. The Jupyter notebook combines two components:

A web application: A browser-based editing program for interactive authoring of computational notebooks which provides a fast interactive environment for prototyping and explaining code, exploring and visualizing data, and sharing ideas with others

Main features of the web application

- In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.
- The ability to execute code from the browser, with the results of computations attached to the code which generated them.
- Displaying the result of computation using rich media representations, such as HTML, LaTeX, PNG, SVG, etc. For example, publication-quality figures rendered by the [matplotlib] library, can be included inline.
- In-browser editing for rich text using the [Markdown] markup language, which can provide commentary for the code, is not limited to plain text.
- The ability to easily include mathematical notation within markdown cells using LaTeX, and rendered natively by MathJax.

Notebook documents

Notebook documents contains the inputs and outputs of a interactive session as well as additional text that accompanies the code but is not meant for execution. In this way, notebook files can serve as a complete computational record of a session, interleaving executable code with explanatory text, mathematics, and rich representations of resulting objects. These documents are internally JSON files and are saved with the .ipynb extension. Since JSON is a plain text format, they can be version-controlled and shared with colleagues. Notebooks may be exported to a range of static formats, including HTML (for example, for blog posts), reStructuredText, LaTeX, PDF, and slide shows, via the [nbconvert] command.

Furthermore, any .ipynb notebook document available from a public URL can be shared via the Jupyter Notebook Viewer <nbviewer>. This service loads the notebook document from the URL and renders it as a static web page. The results may thus be shared with a colleague, or as a public blog post, without other users needing to install the Jupyter notebook themselves. In effect, nbviewer is simply [nbconvert] as a web service, so you can do your own static conversions with nbconvert, without relying on nbviewer.

4.4 SYSTEM ARCHITECTURE

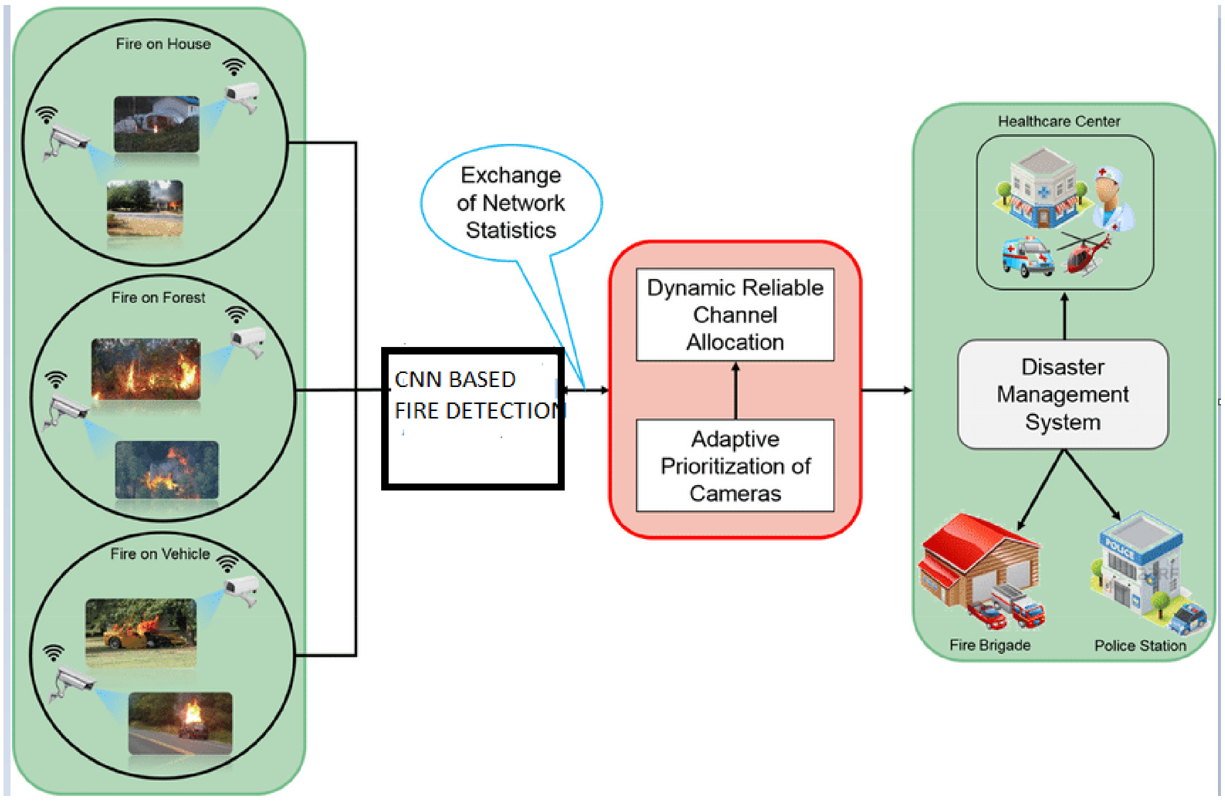

Fire on House
Fire on Forest
Fire on Vehicle
Exchange of Network Statistics
CNN BASED FIRE DETECTION
Dynamic Reliable Channel Allocation
Adaptive Prioritization of Cameras
Healthcare Center
Disaster Management System
Fire Brigade
Police Station

Fig 4.1 System Architecture

Fire detection in the context of disaster management systems during surveillance of public areas, forests, and nuclear power plants, can result in saving of ecological, economical, and social damages. However, early detection is one of the challenging problems due to varying lighting conditions, shadows, and movement of firecolored objects. Thus, there is a need for such an algorithm which can achieve better accuracy in the aforementioned scenarios while minimize the number of false alarms. To achieve this goal, we explored deep CNNs and devised a fine-tuned architecture for early fire detection during surveillance for effective disaster management systems. After successful fire detection, another desirable requirement is sending an immediate alert to disaster management system along with the representative keyframes.

4.5 FLOW CHART

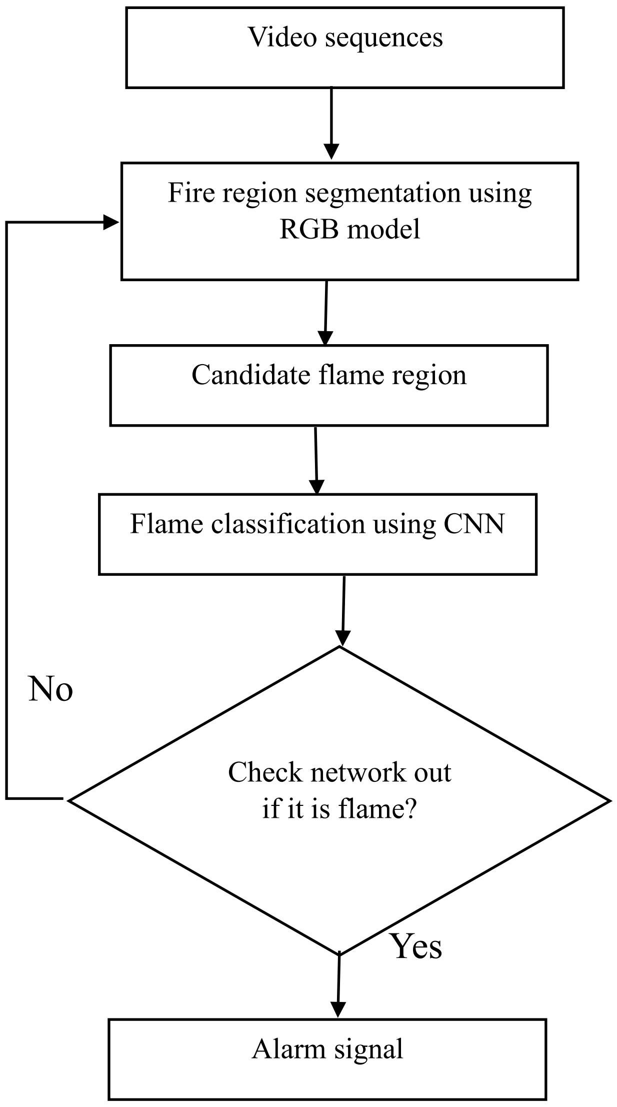

Video sequences
Fire region segmentation using RGB model
Candidate flame region
Flame classification using CNN
Check network out if it is flame?
No
Yes
Alarm signal

Fig 4.2 Flow Chart

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 SYSTEM MODULES

- Dataset collection
- Input layer
- Convolution layer
- Pooling layer
- Fully connected layer

5.2 MODULE DESCRIPTION

5.2.1 Dataset Collection

Dataset is collected from Kaggle dataset. Wildfire dataset, capturing a broad array of environmental conditions, forest types, geographical regions, and confounding elements, aiming to reduce high false alarm rates in fire detection systems. To ensure integrity only public domain images were included, and a detailed description of the dataset's attributes, URL sources and image resolutions is provided.

5.2.2 Input layer

The main task of the Input layer is to preprocess the original image. AlexNet requires an input size of 227 × 227. However, because the sample set in this project was collected through different channels, the size of the sample images is not consistent. Therefore, to reduce the computational complexity, all images were resized to match the input size.

5.2.3 Convolution layer

This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM). The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image. The convolution layer in CNN passes the result to the next layer once applying the convolution operation in the input. Convolutional layers in CNN benefit a lot as they ensure the spatial relationship between the pixels is intact.

5.2.4 Pooling layer

The pooling layer is usually followed by the convolution layer; it is used to reduce the size of the matrix, preserve the main features while reducing the parameters of the next layer, and reduce the computational complexity to prevent overfitting. Max pooling and average pooling methods are used often. For image recognition, the max pooling method can reduce the mean shift caused by convolutional layer parameter errors. It can retain more texture information, which is also important in image processing.

There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.

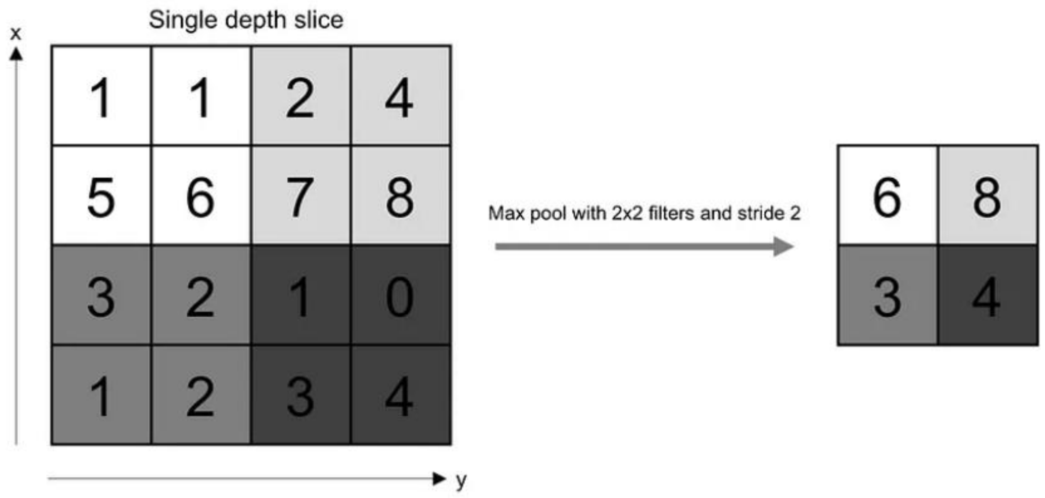


Fig 5.1 Pooling operation

5.2.4 Fully connected layer

The fully connected layer correctly classifies images. To identify whether the target is a flame, the fully connected layer is divided into two categories, 0 and 1, which respectively represent non-fire and fire source images. The number of neurons input to the fully connected layer is greatly reduced through the processing of convolutional and pooling layers. For example, in AlexNet, after processing an image with a size of 227 × 227 and a color channel count of 3, the number of neurons input into the fully connected layer is 4096. Finally, the output of softmax can be determined based on the actual number of classification labels. In the fully connected layer, a dropout mechanism randomly deletes some neurons; this can save time in preventing the overfitting of contracts.

Non-Linearity Layers

Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map.

There are several types of non-linear operations, the popular ones being:

1. Sigmoid

The sigmoid non-linearity has the mathematical form $\sigma(\kappa) = 1/(1+e^{-}\kappa)$. It takes a real-valued number and "squashes" it into a range between 0 and 1. However, a very undesirable property of sigmoid is that when the activation is at either tail, the gradient becomes almost zero. If the local gradient becomes very small, then in backpropagation it will effectively "kill" the gradient. Also, if the data coming into the neuron is always positive, then the output of sigmoid will be either all positives or all negatives, resulting in a zig-zag dynamic of gradient updates for weight.

2. Tanh

Tanh squashes a real-valued number to the range [-1, 1]. Like sigmoid, the activation saturates, but — unlike the sigmoid neurons — its output is zero centered.

3. ReLU

The Rectified Linear Unit (ReLU) has become very popular in the last few years. It computes the function $f(\kappa)=\max(0,\kappa)$. In other words, the activation is simply threshold at zero. In comparison to sigmoid and tanh, ReLU is more reliable and accelerates the convergence by six times.

CHAPTER 6

SYSTEM TESTING

FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS:

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountere

CHAPTER 7

RESULT AND DISCUSSION

The aim of this work is to present a method that can be smoothly deployed to an embedded device in order to finally build a complete fire detection unit. Therefore, it becomes inevitable to use a test dataset that includes images that are often encountered in real-world fire emergencies with an Image quality that is commonly obtained with a camera. To avoid the instances of false alarms being triggered, a threshold for the classifier confidence was set. Hence, the alarm is only triggered when the confidence is greater or equal to the threshold. Our aim is to detect a fire from the video stream with very high accuracy and trigger an alert as quickly as possible. To boost the speed of the classifier, TensorFlow's "optimize_for_inference" script was used to remove all unnecessary nodes in the module. The script also does a few other optimization processes like normalizing operations into the convolutional weights that help speed up the model

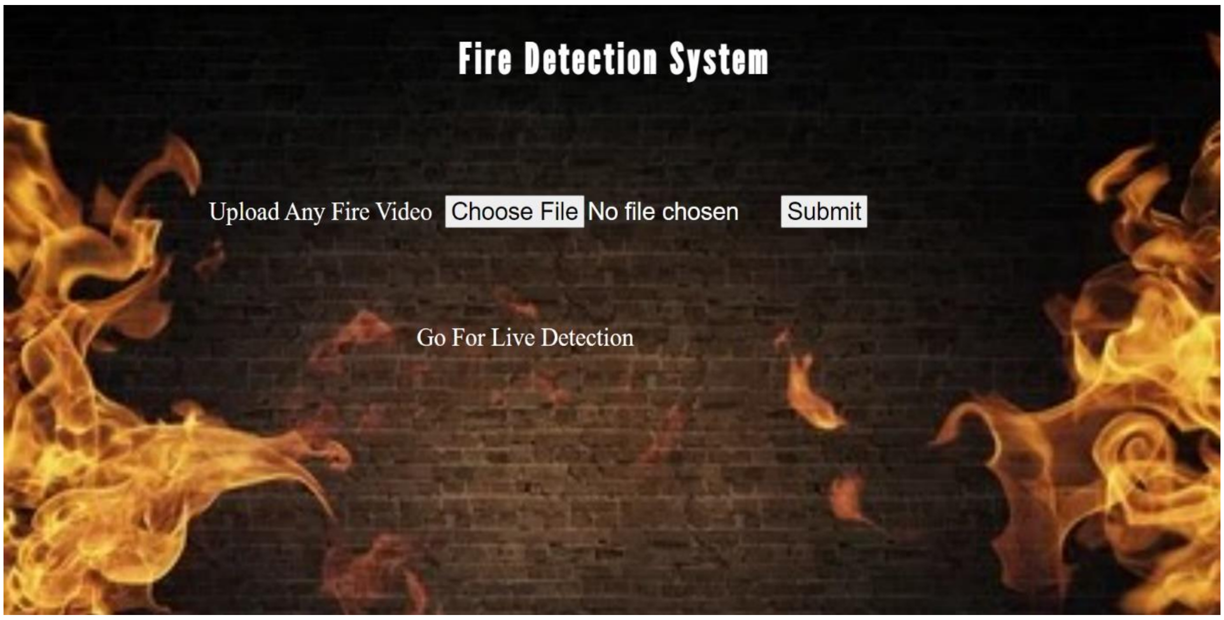


Fig 7.1 Home page

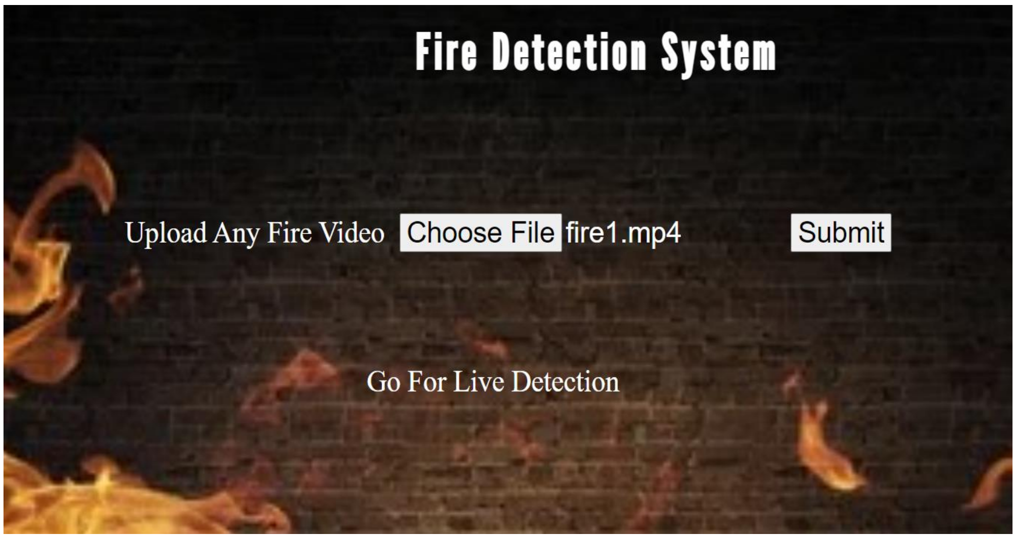


Fig 7.2 Upload dataset

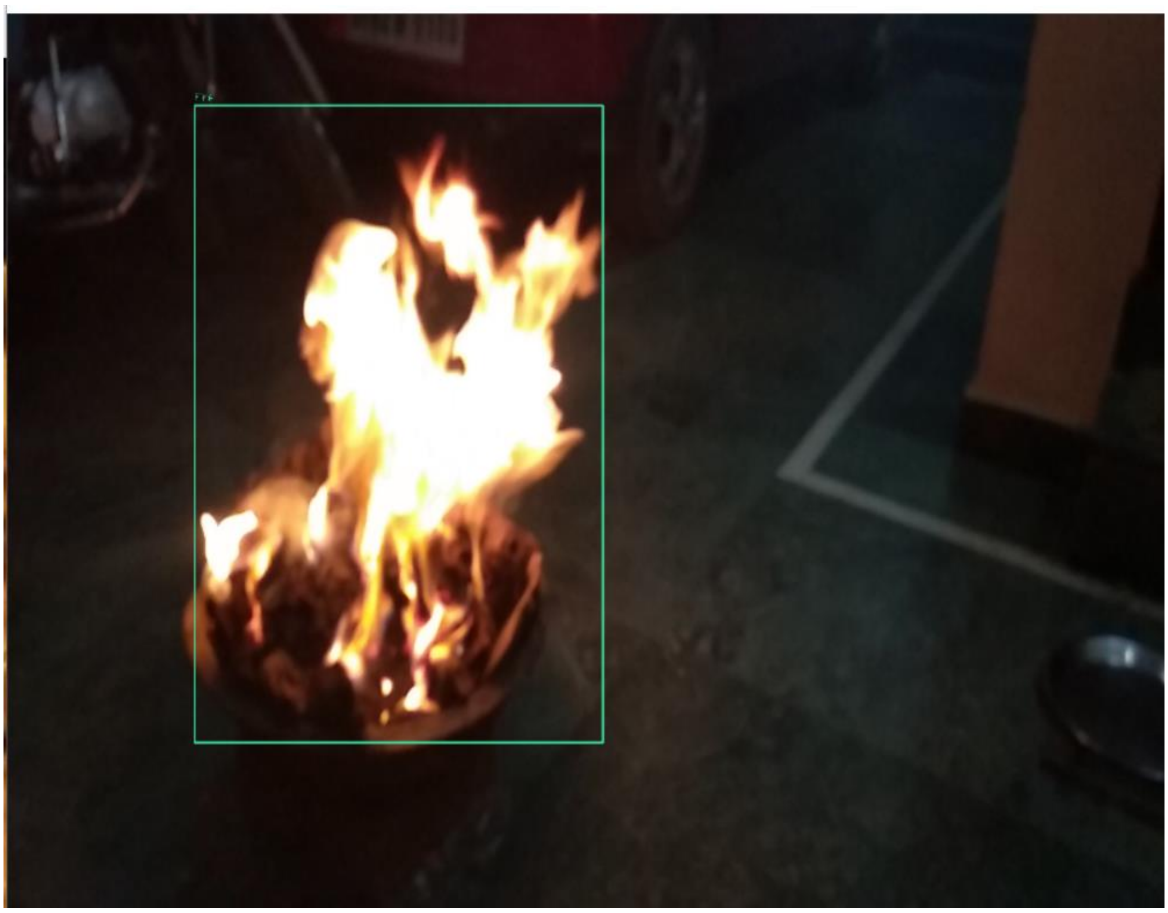


Fig 7.3 Fire detection

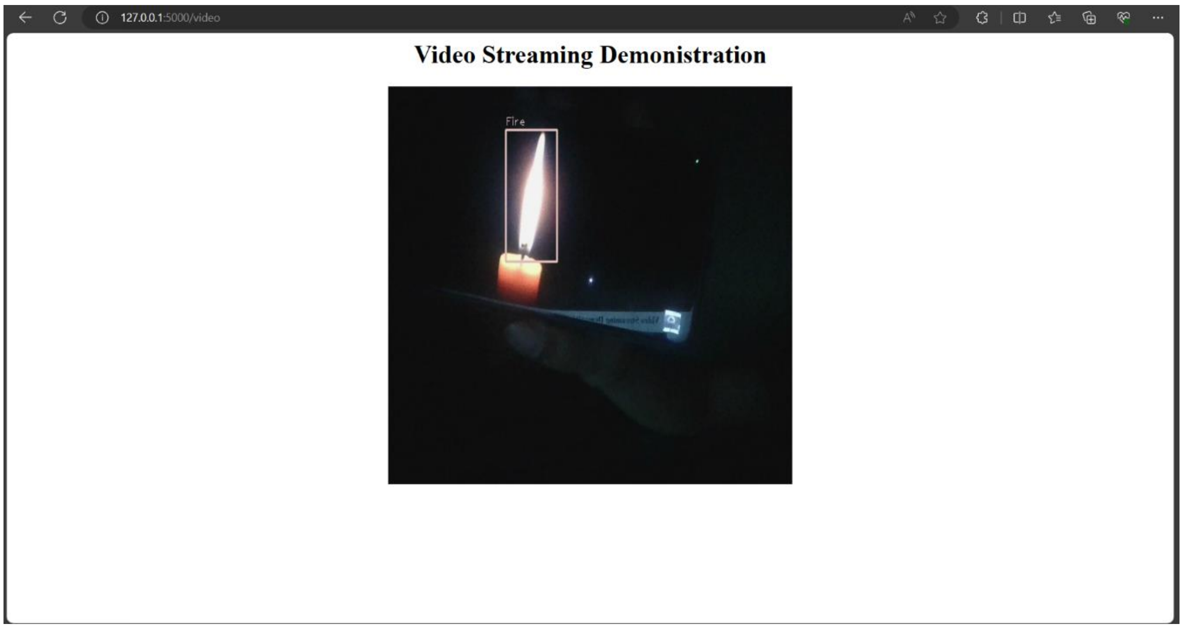
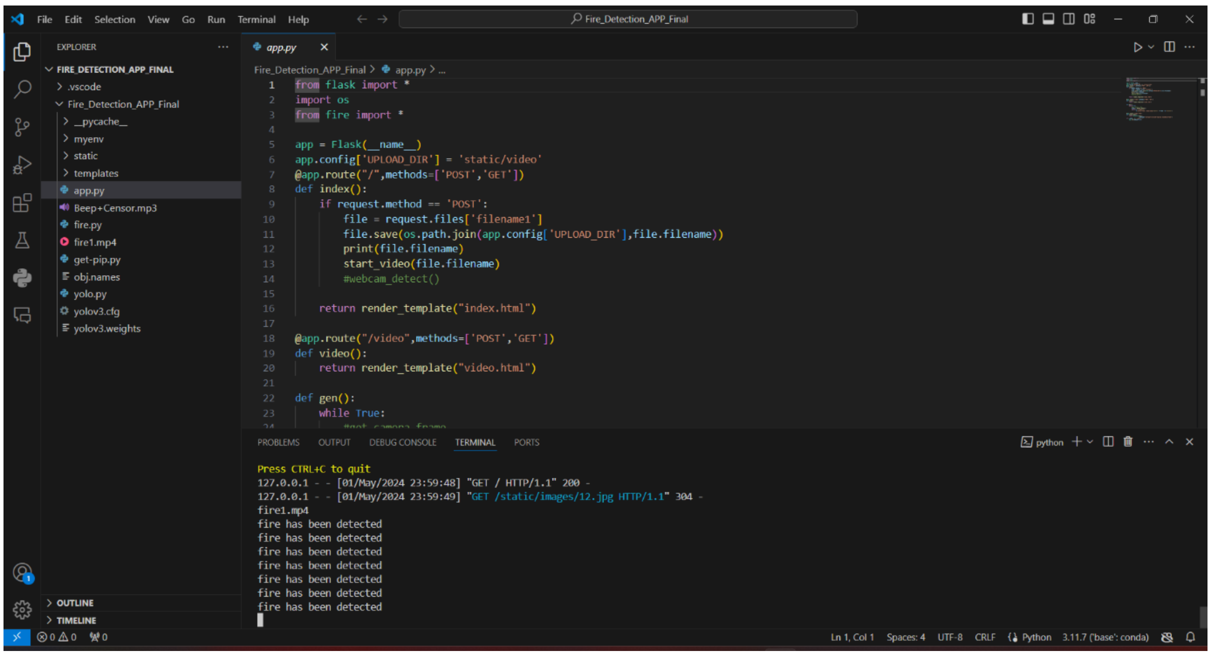


Fig 7.4 Live Video Detection



```python
from flask import *
import os
from fire import *

app = Flask(__name__)
app.config['UPLOAD_DIR'] = 'static/video'
@app.route("/",methods=['POST','GET'])
def index():
    if request.method == 'POST':
        file = request.files['filename1']
        file.save(os.path.join(app.config['UPLOAD_DIR'],file.filename))
        print(file.filename)
        start_video(file.filename)
        #webcam_detect()

    return render_template("index.html")

@app.route("/video",methods=['POST','GET'])
def video():
    return render_template("video.html")

def gen():
    while True:
```

```
Press CTRL+C to quit
127.0.0.1 - - [01/May/2024 23:59:48] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2024 23:59:49] "GET /static/images/12.jpg HTTP/1.1" 304 -
fire1.mp4
fire has been detected
fire has been detected
fire has been detected
fire has been detected
fire has been detected
fire has been detected
fire has been detected
```

Fig 7.5 Output Detection

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

In this project, the process of the forest fire recognition algorithm based on CNN is presented. Its main feature is that the flame image is employed for training and testing. Then, AlexNet model is introduced, and an adaptive pooling method combined with color features is proposed for the problem that the traditional pooling method in CNN may weaken the image features in some cases. The effects of learning rate, batch size, and other parameters on the performance of CNN are analyzed based on experiments, and the optimal parameters are determined. Candidate flame area is extracted based on color feature; thus, the image feature of non-flame area in the hidden layer is reduced and the feature, such as shape and texture is enhanced. The information loss of image are avoided as adaptive pooling is adopted, and the rate of flame recognition in which fire area is segmentation than that of original image is adopted without segmentation.

8.2 FUTURE WORK

It is shown that the proposed algorithm has high recognition rate and is feasible. In this project the pooling of CNN is modified and applied on forest image recognition, recognition rate and consuming time will be developed deeply and compared with other algorithms in future.

APPENDIX

SAMPLE CODING

App.py

```
from flask import *
import os
from fire import *

app = Flask(__name__)
app.config['UPLOAD_DIR'] = 'static/video'
@app.route("/",methods=['POST','GET'])
def index():
    if request.method == 'POST':
        file = request.files['filename1']
        file.save(os.path.join(app.config['UPLOAD_DIR'],file.filename))
        print(file.filename)
        start_video(file.filename)
        #webcam_detect()

    return render_template("index.html")

@app.route("/video",methods=['POST','GET'])
def video():
    return render_template("video.html")

def gen():
    while True:
        #get camera frame
        frame = webcam_detect()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    return Response(gen(),
                    mimetype='multipart/x-mixed-replace; boundary=frame')
if __name__ == '__main__':
    app.run(debug=False)
```

fire.py

```
import cv2
import numpy as np
import argparse
import time
import winsound
parser = argparse.ArgumentParser()
parser.add_argument('--webcam', help="True/False", default=False)
parser.add_argument('--play_video', help="Tue/False", default=False)
parser.add_argument('--image', help="Tue/False", default=False)
parser.add_argument('--video_path', help="Path of video file",
default="videos/fire1.mp4")
parser.add_argument('--image_path', help="Path of image to detect objects",
default="Images/bicycle.jpg")
parser.add_argument('--verbose', help="To print statements", default=True)
args = parser.parse_args()
path1 = r"static\video"
```

```
#Load yolo
def load_yolo():
	net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
	classes = []
	with open("obj.names", "r") as f:
		classes = [line.strip() for line in f.readlines()]

	layers_names = net.getLayerNames()
	output_layers = [layers_names[i-1] for i in
net.getUnconnectedOutLayers()]
	colors = np.random.uniform(0, 255, size=(len(classes), 3))
	return net, classes, colors, output_layers

def load_image(img_path):
	# image loading
	img = cv2.imread(img_path)
	img = cv2.resize(img, None, fx=0.4, fy=0.4)
	height, width, channels = img.shape
	return img, height, width, channels

def start_webcam():
	cap = cv2.VideoCapture(0)

	return cap



def display_blob(blob):
	'''
```

```
        Three images each for RED, GREEN, BLUE channel
    '''
    for b in blob:
        for n, imgb in enumerate(b):
            cv2.imshow(str(n), imgb)


def detect_objects(img, net, outputLayers):
    blob = cv2.dnn.blobFromImage(img, scalefactor=0.00392, size=(320,
320), mean=(0, 0, 0), swapRB=True, crop=False)
    net.setInput(blob)
    outputs = net.forward(outputLayers)
    return blob, outputs


def get_box_dimensions(outputs, height, width):
    boxes = []
    confs = []
    class_ids = []
    for output in outputs:
        for detect in output:
            scores = detect[5:]
            class_id = np.argmax(scores)
            conf = scores[class_id]
            if conf > 0.3:
                center_x = int(detect[0] * width)
                center_y = int(detect[1] * height)
                w = int(detect[2] * width)
                h = int(detect[3] * height)
                x = int(center_x - w/2)
```

```
                    y = int(center_y - h / 2)
                    boxes.append([x, y, w, h])
                    confs.append(float(conf))
                    class_ids.append(class_id)
        return boxes, confs, class_ids
def sound():
    import winsound
    #Beep at frequency = 5000 Hz for duration of 1000 ms
    winsound.Beep(5000, 1000)
    #windows exit sound after completion of above
    #winsound.PlaySound("SystemExit", winsound.SND_ALIAS)

def draw_labels(boxes, confs, colors, class_ids, classes, img):
        indexes = cv2.dnn.NMSBoxes(boxes, confs, 0.5, 0.4)
        font = cv2.FONT_HERSHEY_PLAIN
        for i in range(len(boxes)):
                if i in indexes:
                        x, y, w, h = boxes[i]
                        label = str(classes[class_ids[i]])

                        color = colors[i]
                        cv2.rectangle(img, (x,y), (x+w, y+h), color, 2)
                        cv2.putText(img, label, (x, y - 5), font, 1, color, 1)
                        if label == 'Fire':
                                print('fire and smoke detected')
                                sound()
                                break
                else:
                        pass
        img=cv2.resize(img, (800,600))
        # cv2.imshow("Image", img)
def draw_labels1(boxes, confs, colors, class_ids, classes, img):
        indexes = cv2.dnn.NMSBoxes(boxes, confs, 0.5, 0.4)
        font = cv2.FONT_HERSHEY_PLAIN
        for i in range(len(boxes)):
                if i in indexes:
                        x, y, w, h = boxes[i]
                        label = str(classes[class_ids[i]])

                        color = colors[i]
                        cv2.rectangle(img, (x,y), (x+w, y+h), color, 2)
                        cv2.putText(img, label, (x, y - 5), font, 1, color, 1)
                        if label == 'Fire':
                                print('fire has been detected')
                                sound()
                                break
                        else:
                                pass
        img=cv2.resize(img, (800,600))
        cv2.imshow("Image", img)

def image_detect(img_path):
        model, classes, colors, output_layers = load_yolo()
        image, height, width, channels = load_image(img_path)
        blob, outputs = detect_objects(image, model, output_layers)
        boxes, confs, class_ids = get_box_dimensions(outputs, height, width)
        draw_labels(boxes, confs, colors, class_ids, classes, image)
        while True:
                key = cv2.waitKey(1)
                if key == 27:
                        break

def webcam_detect():
        model, classes, colors, output_layers = load_yolo()
        cap = start_webcam()
        while True:
                _, frame = cap.read()
                height, width, channels = frame.shape
                blob, outputs = detect_objects(frame, model, output_layers)
                boxes, confs, class_ids = get_box_dimensions(outputs, height,
width)
                draw_labels(boxes, confs, colors, class_ids, classes, frame)
                key = cv2.waitKey(1)
                if key == 27:
                        break
                _, jpeg = cv2.imencode('.jpg', frame)
                return jpeg.tobytes()
        # cap.release()



def start_video(video_name):
        video_path = path1+ '/'+ video_name
```

```
	model, classes, colors, output_layers = load_yolo()
	cap = cv2.VideoCapture(video_path)

	while True:
		_, frame = cap.read()
		height, width, channels = frame.shape
		blob, outputs = detect_objects(frame, model, output_layers)
		boxes, confs, class_ids = get_box_dimensions(outputs, height, 
width)
		draw_labels1(boxes, confs, colors, class_ids, classes, frame)

		key = cv2.waitKey(1)
		if cv2.waitKey(1) & 0xFF ==ord('q'):
			break
	cap.release()

if __name__ == '__main__':
	webcam = args.webcam
	video_play = args.play_video
	image = args.image
	if webcam:
		if args.verbose:
			print('---- Starting Web Cam object detection ----')
		webcam_detect()
	if video_play:
		video_path = args.video_path
		if args.verbose:
			print('Opening '+video_path+" .... ")
```

```
            start_video(video_path)
        if image:
            image_path = args.image_path
            if args.verbose:
                print("Opening "+image_path+" .... ")
            image_detect(image_path)
        cv2.destroyAllWindows()
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Anton&family=League+Gothic
&display=swap" rel="stylesheet">
   <title>Fire Detection System</title>
   <style>
      body{
         background-image: url('static/images/12.jpg');
         background-size: cover;
         background-repeat: no-repeat;
      }
      h1{
         color: white;
```

```
    text-align: center;
    font-family: 'Anton', sans-serif;
    font-family: 'League Gothic', sans-serif;
    letter-spacing: 5px;
    font-size: 50px;
}
label,.upload{
    color: white;
    top: 100px;
    left: 250px;
    position: relative;
    font-size: 30px;
}
.upload1{
    top: 100px;
    left: 100px;
    position: relative;
    font-size: 30px;
}
.upload2{
    top: 250px;
    right: 470px;
    position: relative;
    font-size: 30px;
    text-decoration: none;
    color: white;
}
```

```
   </style>
</head>
<body>
   <h1>Fire Detection System</h1>

   <form action="/" method="POST" enctype="multipart/form-data">
       <label>Upload Any Fire Video</label>
       &nbsp;&nbsp;&nbsp;<input class="upload" type="file"
name="filename1">
       <input class="upload1" type="submit">
       <a class="upload2" href="{{url_for('video')}}">Go For Live
Detection</a>
       <!-- <input class="upload2" value="Go For Live Detection"
type="submit"> -->
   </form>

</body>
</html>
```

Video_upload.html

```
<html>
  <head>
   <title>Fire Detection</title>
   <style>
         img {
         display: block;
         margin-left: auto;
         margin-right: auto;
         width: 35%
         }
    </style>
  </head>
  <body>
    <h1 align="center">Fire Detection System</h1>
    <img id="bg" class="center" src="{{ url_for('video_feed1') }}"
width="700px" height="500px">
  </body>
</html>
```

Video.html

```
<html>
  <head>
    <title>Live Detection</title>
    <style>
         img {
         display: block;
         margin-left: auto;
         margin-right: auto;
         width: 35%
         }
    </style>
  </head>
  <body>
    <h1 align="center">Video Streaming Demonistration</h1>
    <img id="bg" class="center" src="{{ url_for('video_feed') }}"
width="700px" height="500px">
  </body>
</html>
```

REFERENCES

[1] A. Verhegghen et al., “The potential of Sentinel satellites for burnt area mapping and monitoring in the Congo Basin forests”, Remote Sens., vol. 8, no. 12, pp. 1–22, 2016.

[2] C. O. Justice et al., “The MODIS fire products”, Remote Sens. Environ., vol. 83, nos. 1/2, pp. 244–262, 2002.

[3] D. Thomas, D. Butry, S. Gilbert, D. Webb, and J. Fung, “The costs and losses of wildfires: A literature survey (NIST Special Publication 1215)”, 2017. [Online]. Available: https://doi.org/10.6028/NIST.SP.1215

[4] E. Chuvieco et al., “Historical background and current developments for mapping burned area from satellite Earth observation”, Remote Sens. Environ., vol. 225, pp. 45–64, 2019.

[5] J. A. Cardille and J. A. Fortin, “Bayesian updating of land-cover estimates in a data-rich environment”, Remote Sens. Environ., vol. 186, pp. 234–249, 2016.

[6] J. Wang et al., “Review of satellite remote sensing use in forest health studies”, Open Geography J., vol. 3, no. 1, pp. 28–42, 2010.

[7] K. Lasko etal, “Incorporating Sentinel-1 SAR imagery with the MODIS MCD64A1 burned area product to improve burn date estimates and reduce burn date uncertainty in wildland fire mapping”, Geocarto Int., vol. 36, no. 3, pp. 340–360, 2019.

[8] M. Caggiano, “Using community base maps to improve the safety and effectiveness of wildfire response”, 2018. Accessed: Oct. 7, 2020. [Online]. Available: https://fireadaptednetwork.org/using-community-base-mapsto-improve-the-safety-and-effectiveness-of-wildland-fire-response/

[9] P. Jain, S. C. Coogan, S. G. Subramanian, M. Crowley, S. W. Taylor, and M. D. Flannigan, “A review of machine learning applications in wildfire science and management”, Environ. Rev., vol. 28, pp. 1–70, 2020.
[10] S. C. Coogan, F. N. Robinne, P. Jain, and M. D. Flannigan, “Scientists’ warning on wildfire—A Canadian perspective”,Can. J. Forest Res., vol. 49, no. 9, pp. 1015–1023, 2019.
[11] Z. Langford, J. Kumar, and F. Hoffman, “Wildfire mapping in interior Alaska using deep neural networks on imbalanced datasets”, in Proc. IEEE Int. Conf. Data Mining Workshops, 2018, pp. 770–778.