



EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES



IBM NALAIYATHIRAN

PROJECT REPORT

Submitted By

DHINAKARAN K (611219106015)

REVATHY V (611219106062)

SANTHOSH S (611219106063)

SHYAM V (611219106068)

in partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

in

**ELECTRONICS AND COMMUNICATION
ENGINEERING**

KNOWLEDGE INSTITUTE OF TECHNOLOGY,

SALEM-637504

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
1	INTRODUCTION	1
	1.1 Project Overview	1
	1.2 Purpose	1
2	LITERATURE SURVEY	2
	2.1 Existing Problems	2
	2.2 Problem Statement Definition	4
3	IDEATION AND PROPOSED SOLUTION	5
	3.1 Empathy Map	5
	3.2 Ideation and Brainstorming	6
	3.3 Proposed Solution	9
	3.4 Problem Solution Fit	10
4	REQUIREMENT ANALYSIS	11
	4.1 Functional Requirement	11
	4.2 Non-Functional Requirement	11
5	PROJECT DESIGN	12
	5.1 Data Flow Diagrams	12
	5.2 Solution and Technical Architecture	13
	5.3 User Stories	14

6	PROJECT PLANNING AND SCHEDULING	15
	6.1 Sprint Planning and Estimation	15
	6.2 Sprint Delivery Schedule	17
	6.3 Reports from JIRA	17
7	CODING AND SOLUTION	19
	7.1 Feature 1	19
	7.2 Feature 2	20
8	TESTING	21
	8.1 Test Cases	21
	8.2 User Acceptance Testing	23
9	RESULTS	24
	9.1 Performance Metrics	24
10	ADVANTAGES AND DISADVANTAGES	25
11	CONCLUSION	26
12	FUTURE SCOPE	27
13	APPENDIX	28
	13.1 Source Code	28
	13.2 GitHub and Project Demo Link	32
14	REFERENCES	33

LIST OF TABLES

Table No.	Table Name	Page No.
4.1	Functional Requirements	11
4.2	Non-Functional Requirements	11
5.3	User Stories	14
6.2	Sprint Delivery Schedule	17
8.1.1	Test Cases	21
8.1.2	Test Report	22
8.2	Test Case Analysis	23

LIST OF FIGURES

Figure No.	Figure Name	Page No.
2.2	Problem Statement Definition	4
3.1	Empathy Map	5
3.2.1	Defining Problem Statement	6
3.2.2	Brainstorming and Idea Grouping	7
3.2.3	Idea Prioritization	8
3.4	Problem Solution Fit	10
5.1	Data Flow	12
5.2.1	Solution Architecture	13
5.2.2	Technical Architecture	13
6.3.1	Velocity Report	17
6.3.2	Cumulative Flow Diagram	18
6.3.3	Roadmap	18
7.1	Python Code	19
7.2	HTML and CSS Code	20
9.1.1	Plot depicting accuracy	24
9.1.2	Plot depicting loss	24

LIST OF ABBREVIATION

AI	Artificial Intelligence
CNN	Convolutional Neural Networks
API	Application Programming Interface
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
URL	Uniform Resource Locator
UAV	Unmanned Aerial Vehicle
LoRaWAN	Long Range Wide Area Network

CHAPTER 1

INTRODUCTION

1.1 Project Overview

Natural disasters have always been mankind's constant companion since time immemorial. Forest fire is one such disaster which when occurs at large scale not only destroys the flora, fauna, vegetation of the forest but also puts the life of human being and animals at a very high risk. In the recent past years, managing this type of crisis, viz., a large-scale fire has become a very difficult and challenging task. The proposed methodology involves a deep learning model which learns based upon live camera footage and data to detect forest fires. The deep learning model uses various aspects of deep learning like Convolutional Neural Networks and Artificial Intelligence to get the desired results efficiently and accurately. By using three layers in Convolutional Neural Network, the video frames are classified accordingly. The model is planned to alert local fire authorities and police stations to evacuate and support extinguishing the imminent fire. This project uses Twilio API to provide message alerts. Therefore, a comprehensive survey on the existing forest fire detection and monitoring mechanisms is highly desired. This project has been aimed at utilizing artificial intelligence to monitor, detect and deter forest fires in its starting phase.

1.2 Purpose

Fire can destroy wildlife habitats, homes, timber and polluting the air with emissions harmful to health. Natural causes like high atmospheric temperatures and dryness offer favorable circumstance for a fire to start. To detect forest fire caused by all the above reasons, solutions have been provided in this project.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem

- 1. Early Forest Fire Detection using Drones and Artificial Intelligence (Diyana Kinaneva et al, May 2019)**

Objective:

To detect forest fires early, the proper categorization of fire and fast response from the firefighting departments.

Methodology Used:

The fire detection is based on a platform that uses Unmanned Aerial Vehicles (UAVs) which constantly patrol over potentially threatened by fire areas. The UAVs utilize the benefits from Artificial Intelligence (AI). This allows to use computer vision methods for recognition and detection of smoke or fire, based on images or video input from the drone cameras.

- 2. Emerging methods for early detection of forest fires using Unmanned Aerial Vehicles and LoRaWAN Sensor networks (Georgi Hristov et al, 2018)**

Objective:

To fight forest fires occurring throughout the year with an increasing intensity in the summer and autumn periods.

Methodology Used:

The development of systems for early forest fire detection using LoRaWAN sensor networks and also with the use of a combination between a fixed-wing and a rotary-wing UAVs.

3. Developing a real-time and automatic early warning system for forest fire (A. Sai Chand et al, 2018)

Objective:

To detect forest fires causing by climatic conditions and also caused by human.

Methodology Used:

The method using here is making use of stand-alone boxes which are deployed throughout the forest. Those boxes contain different sensors and a radio module to transmit data received from these sensors. Each sensor will be tested in individually and XBee modules are configured and paired using XCTU Software.

4. Early Fire Detection System using wireless sensor networks (Benamar Kadri et al, November 2018)

Objective:

To detect fires from huge cause of forests.

Methodology Used:

The hierarchical architecture of Wireless Sensor Networks is most efficient and extensible for dense networks which simplifies the management of the forest as well as the communication and the localization of fire and sensors.

5. Automatic Early Forest fire Detection based Gaussian Mixture Model (Jiye Qian et al, 2018)

Objective:

To avoid the huge damage of forest caused by fires.

Methodology Used:

Based on the slow spread of smoke, firstly a time delay parameter improves Gaussian mixture model for extracting candidate smoke regions. Then, two motion features of smoke, the rate of area change and motion style are used to select smoke regions from the candidate regions.

2.2 Problem Statement Definition

Creating a problem statement is to understand the customer's point of view. It helps to focus on what matters to create experiences people will love.

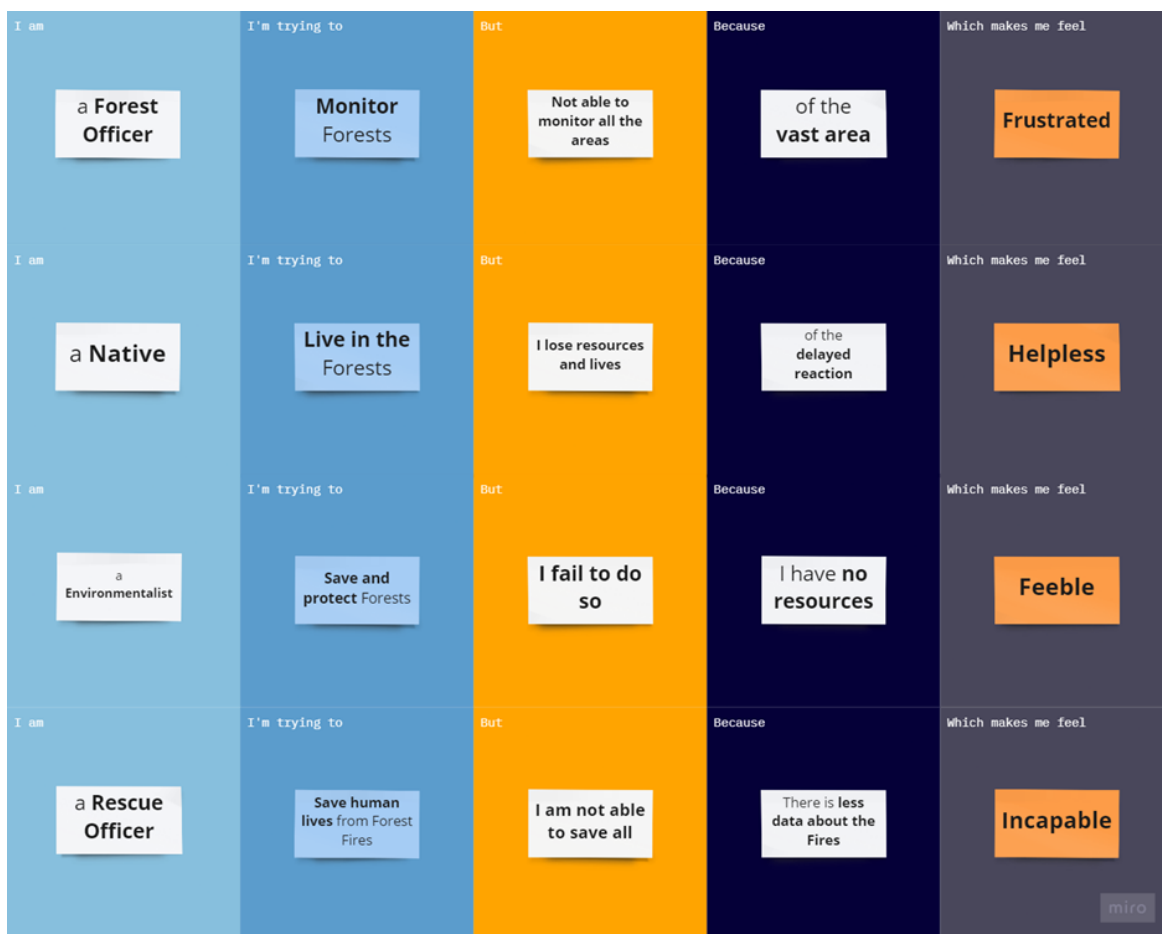


Figure – 2.2: Problem Statement Definition

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

It lists out the thoughts based on various factors like advantages and disadvantages, places to be improved, difficulties faced and the difficulties which are solved.

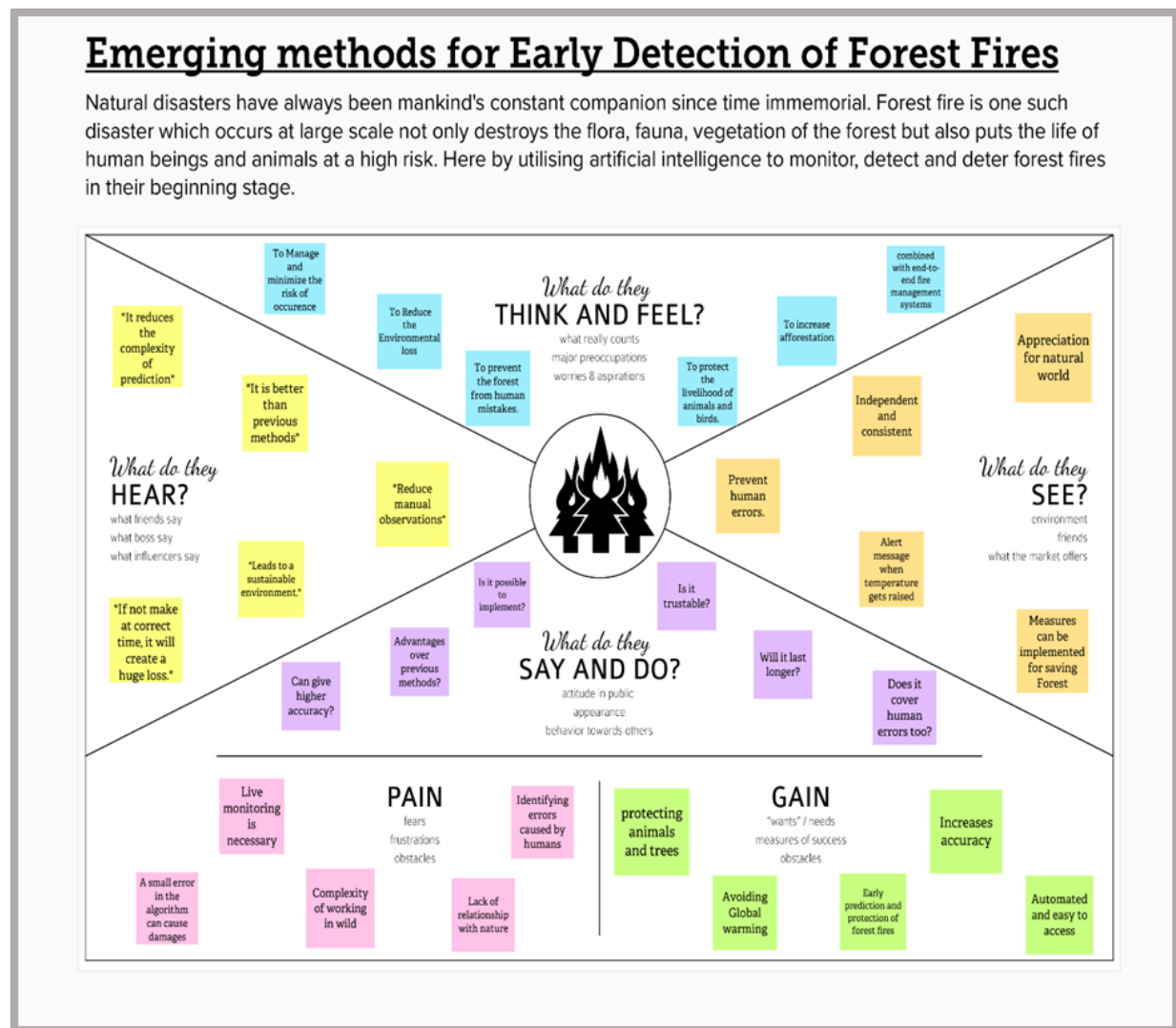


Figure – 3.1: Empathy Map

3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value and all participants have been encouraged to collaborate, help each other to develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

- A Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- B Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.
- C Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.
[Open article](#)

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

Emerging Methods for Early Detection of Forest Fires

Natural disasters have always been mankind's constant companion since time immemorial. Forest fire is one such disaster which occurs at large scale not only destroys the flora, fauna, vegetation of the forest but also puts the life of human beings and animals at a high risk. Here by utilising artificial intelligence to monitor, detect and deter forest fires in their beginning stage.

Technical Architecture

```

graph LR
    Camera[Video Feed From Camera] --> Frames[Frames from Video]
    Frames --> Model[Deep Learning Model]
    Model -- "No Fire Detected" --> Camera
    Model -- "Forest Fire Detected" --> Alert[Alert]
  
```

Figure – 3.2.1: Defining problem statement

Step-2: Brainstorm, Idea Listing and Grouping



Figure – 3.2.2: Brainstorming and Idea Grouping

Step-3: Idea Prioritization



Figure – 3.2.3: Idea prioritization

3.3 Proposed Solution

1. Problem Statement (Problem to be solved):

Forest fires are a major threat to the environment, animals, plants and humans who inhabit them. The current monitoring systems for forests are not equipped with the best for detecting forest. Due to this outdated model there is a delayed response and this delay leads to loss of lives and resources.

2. Idea / Solution Description:

This solution uses Artificial Intelligence (AI) to analyse video and image data of forests to predict and detect forest fires. It also sends alerts to the respective fields for quick evacuation and response to control the fire in advance.

3. Novelty / Uniqueness:

It uses Convolutional Neural Networks which consist of input, hidden and output layers of interconnected neurons. Depending on the number of hidden layers, it utilizes Machine Learning and Deep Learning methods.

4. Social Impact / Customer Satisfaction:

The application helps to provide critical information about the forests and save lives and valuable resources easily when implemented. The people being supported by the forests and the people who support the forests, both are highly benefitted by this solution.

5. Business Model (Revenue model):

This project can help government to create contract with this application to use our tools to protect the forests. This project aims to provide the service, help them maintain the tools and troubleshoot any issues that users face.

6. Scalability of the solution:

The range of the forests can be increased by including sensors for humidity, temperature and other sensors to increase the amount of critical data at hand without compromising the true – false ratio of the results generated by the AI Model.

3.4 Problem Fit Solution

The Problem-Solution Fit simply lists the information whenever a customer finds a problem and the solution has been realized for it, to solve the customer's problem. It helps entrepreneurs, marketers and corporate innovators to identify the behavioral patterns and it also recognizes what and why it works.

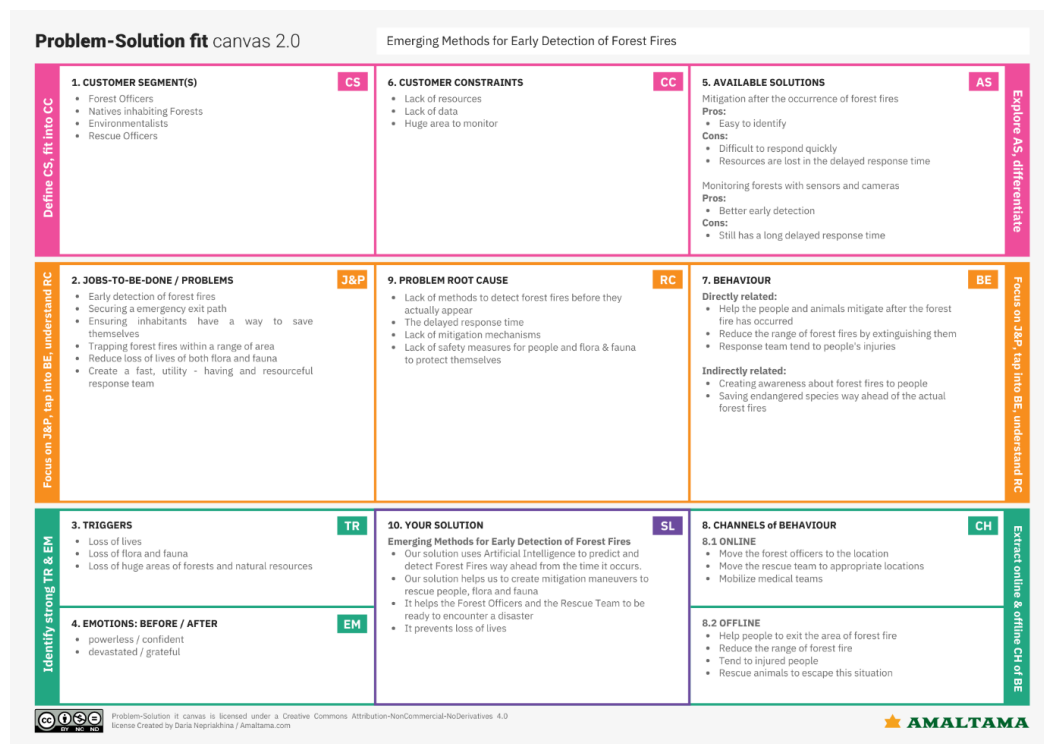


Figure – 3.4: Problem Solution Fit

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirements

Table - 4.1: Functional Requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Monitoring	<ul style="list-style-type: none"> Prediction using Artificial Intelligence Live status of patches of Forests
FR-2	Message Service	<ul style="list-style-type: none"> Twilio Messaging API User Registration and Verification

4.2 Non-Functional Requirements

Table - 4.2: Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	This solution is intended to assist Forest Officers, Environmentalist and Fire Fighters
NFR-2	Security	Used abstraction to hide unnecessary information from the general user.
NFR-3	Reliability	Continuous visual output and if fire is detected it will send an alarm and also, it can be seen in log file.
NFR-4	Performance	Transfer learning approach to improve accuracy and performance of the application.
NFR-5	Availability	Application can work 24/7 when deployed.
NFR-6	Scalability	With the help of cloud services, a greater number of users can access the application.

CHAPTER 5

PROJECT DESIGN

5.1 Data Flow Diagram

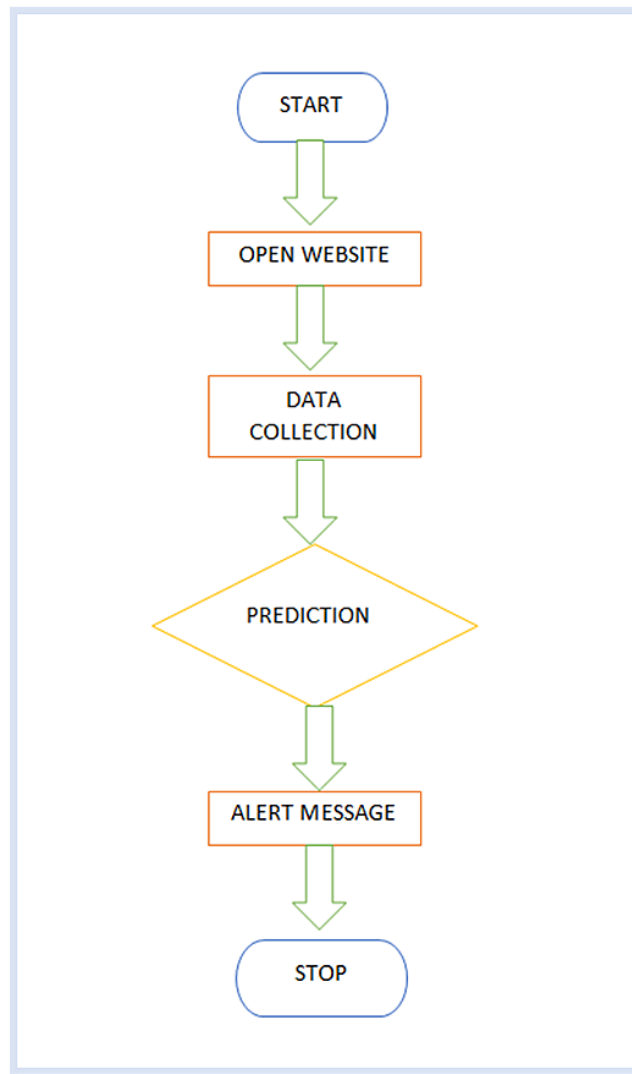


Figure – 5.1: Data Flow

5.2 Solution and Technical Architecture

The solution architecture includes the components and the flow. It has been designed to deliver the solution.

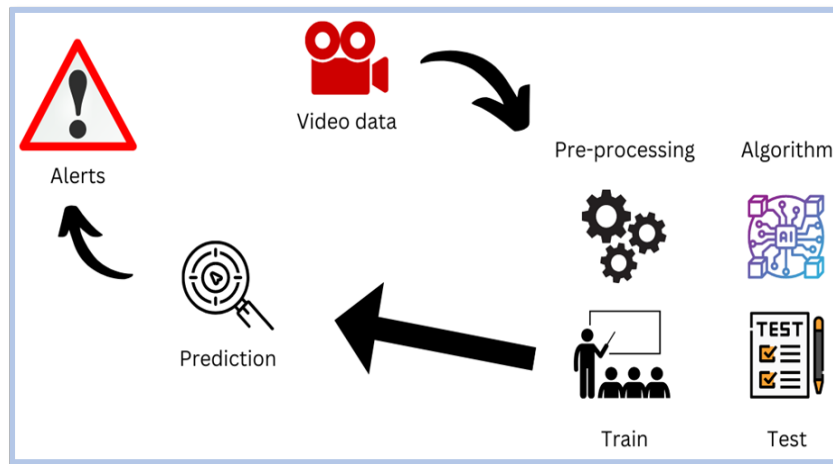


Figure - 5.2.1: Solution Architecture

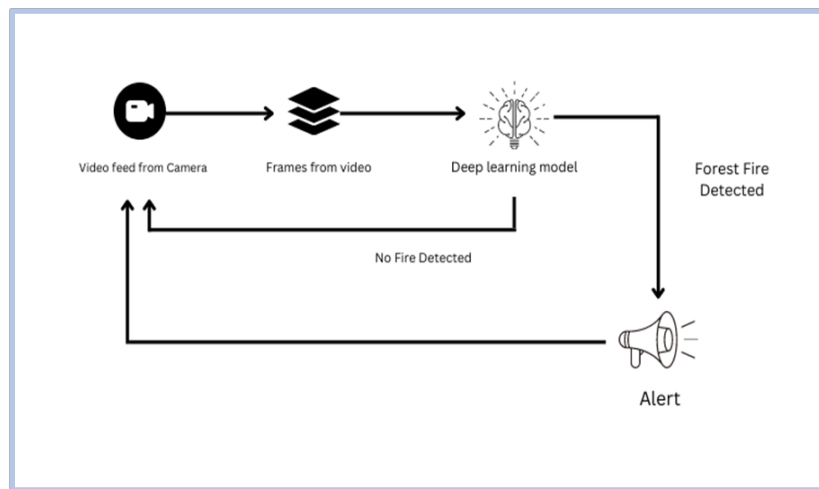


Figure - 5.2.2: Technical Architecture

5.3 User Stories**Table – 5.3: User Stories**

User Type	Functional Requirement (Epic)	User Story / Task	Acceptance criteria	Priority	Release
Forest Officer	Forest Fire detection	As a User, I can install and use the model to detect forest fires.	Model works 24/7	High	Sprint-1
Fire Fighter	Forest Fire Detection	As a User, I can install and use the model to detect forest fires.	Model works 24/7	High	Sprint-1

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning and Estimation

Sprint 1:

Functional Requirement: - Data Collection and Preprocessing

Story Points: - 8

Task 1: - Downloaded data and filtered

Task 2: - Pre-processing measures have been taken for better data

Task 3: - Image data generator

Task 4: - Training and testing data generated

Sprint 2:

Functional Requirement: - Model Building and Integrating Twilio

Story Points: - 10

Task 1: - Model building measures for building better model

Task 2: - Model building initializing and adding layers

Task 3: - Train model with data

Task 4: - Integrating with Twilio

Sprint 3:

Functional Requirement: - Deploying with OpenCV Live Cam

Story Points: - 11

Task 1: - Saving model and loading the OpenCV model

Task 2: - Set OpenCV for live cam

Task 3: - Connecting front-end and back-end

Task 4: - Deploying application using IBM Cloud

Sprint 4:

Functional Requirement: - Testing and Reinforcement

Story Points: - 10

Task 1: - Testing the whole model

Task 2: - Alteration and modifications

Task 3: - Integrating Flask

Task 4: - Final Testing

6.2 Sprint Delivery Schedule

Table – 6.2: Sprint Delivery Schedule

Sprint	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)
Sprint-1	6 Days	24 Oct 2022	29 Oct 2022	29 Oct 2022
Sprint-2	4 Days	30 Oct 2022	02 Nov 2022	02 Nov 2022
Sprint-3	5 Days	03 Nov 2022	07 Nov 2022	07 Nov 2022
Sprint-4	5 Days	08 Nov 2022	12 Nov 2022	12 Nov 2022

6.3 Reports from JIRA

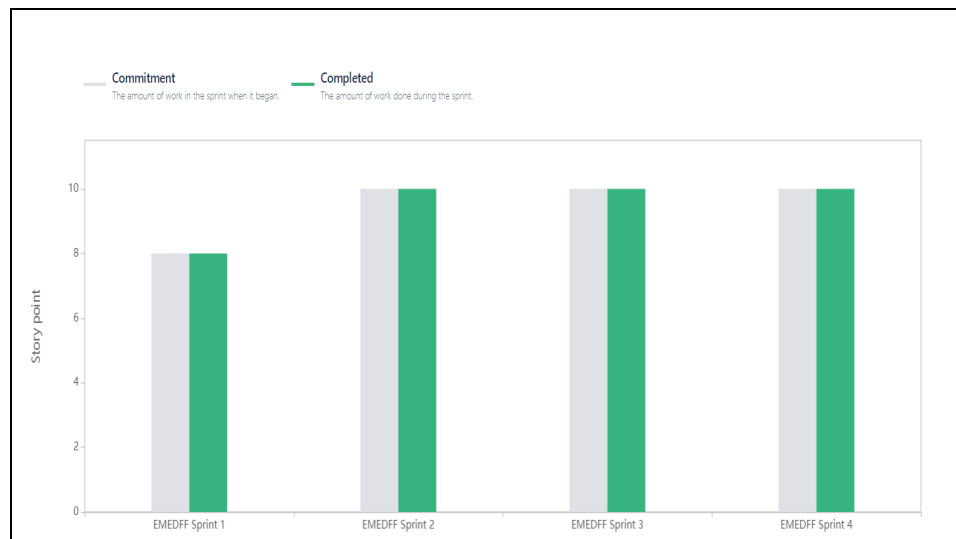


Figure – 6.3.1: Velocity report

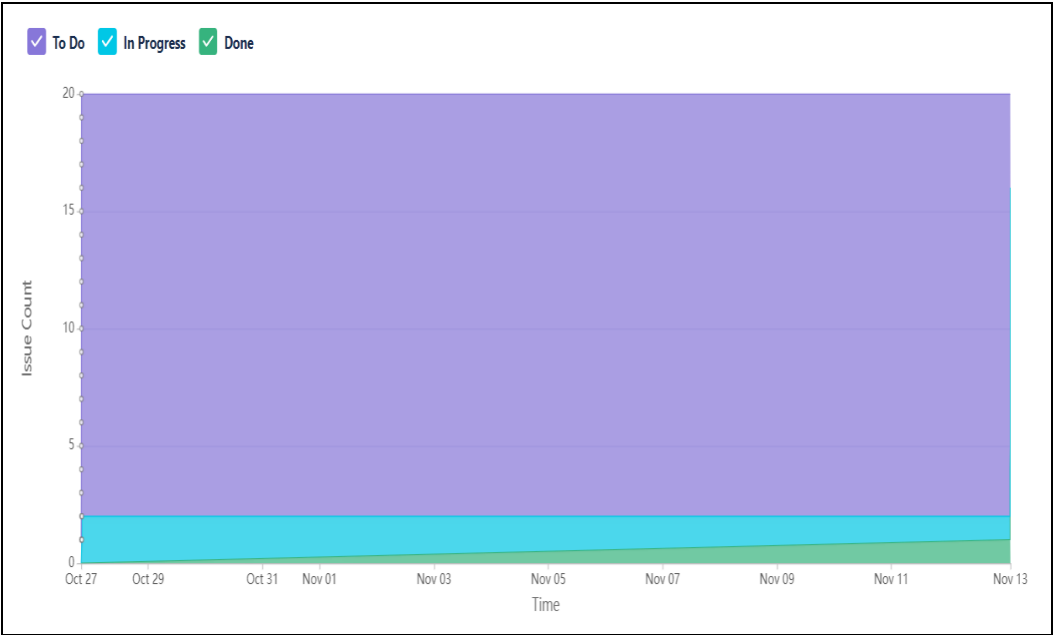


Figure – 6.3.2: Cumulative flow diagram

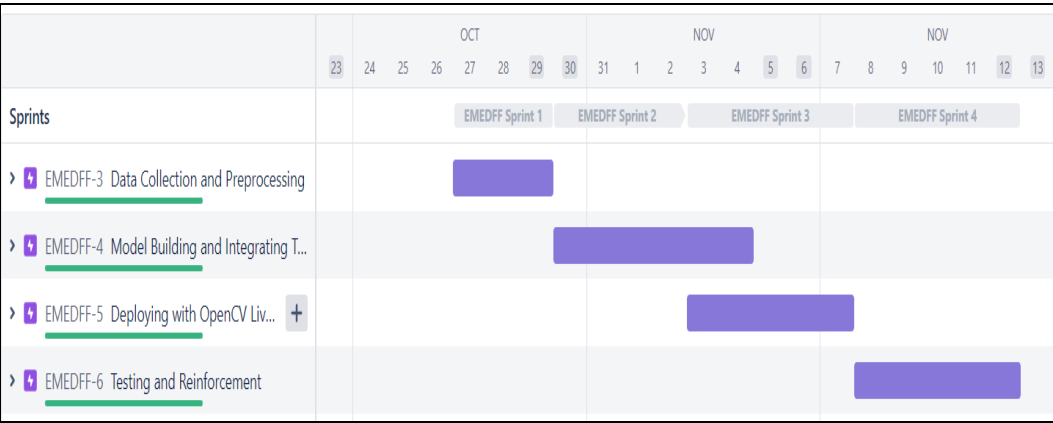


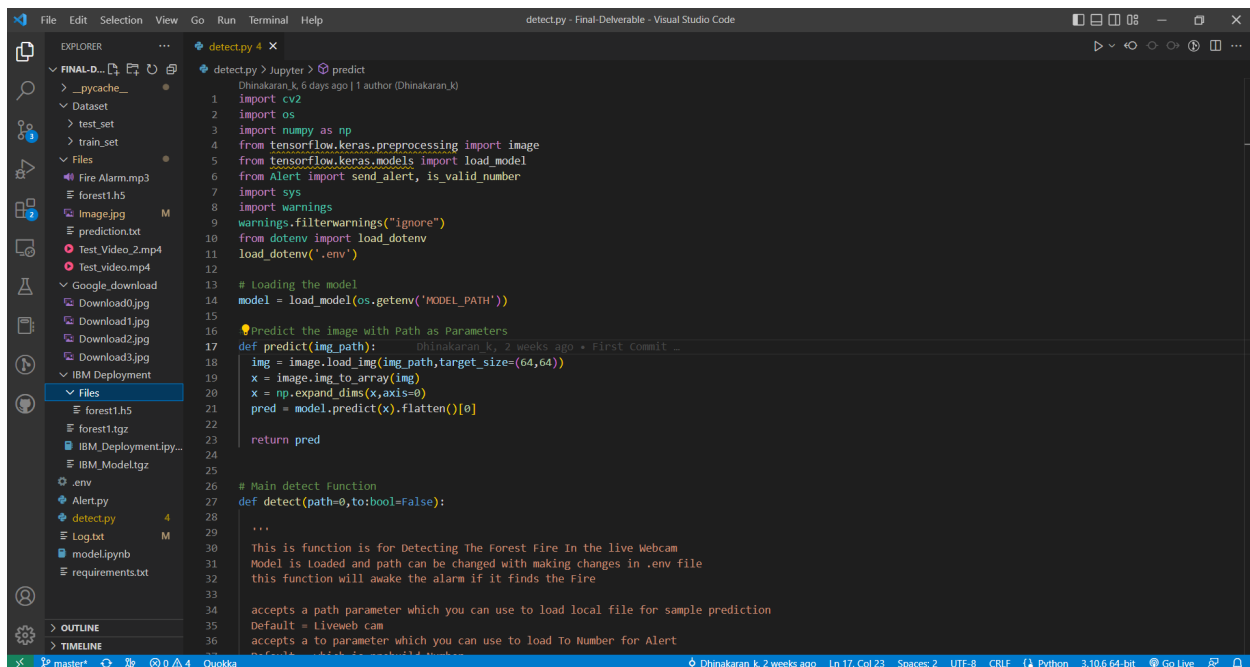
Figure – 6.3.3: Roadmap

CHAPTER 7

CODING AND SOLUTION

7.1 Feature 1

Python is a general-purpose high level programming language that is widely used in data science and for producing deep learning algorithms. This model also used multiple libraries along Python to implement features. Tensorflow and Keras for model building, IBM Cloud for cloud deployment, OpenCV for image detection and also Twilio for sending SMS messages.



```

detect.py - Final Deliverable - Visual Studio Code
EXPLORER
  FINAL-D...
    __pycache__
    Dataset
    test_set
    train_set
    Files
      Fire Alarm.mp3
      forest1.h5
      Image.jpg
      prediction.txt
      Test_Video_2.mp4
      Test_Video.mp4
    Google_download
      Download0.jpg
      Download1.jpg
      Download2.jpg
      Download3.jpg
    IBM Deployment
      Files
        forest1.h5
        forest1.tgz
        IBM_Deployment.Ipy...
        IBM_Model.tgz
        .env
        Alert.py
        detect.py
        Log.txt
        model.ipynb
        requirements.txt
    OUTLINE
    TIMELINE

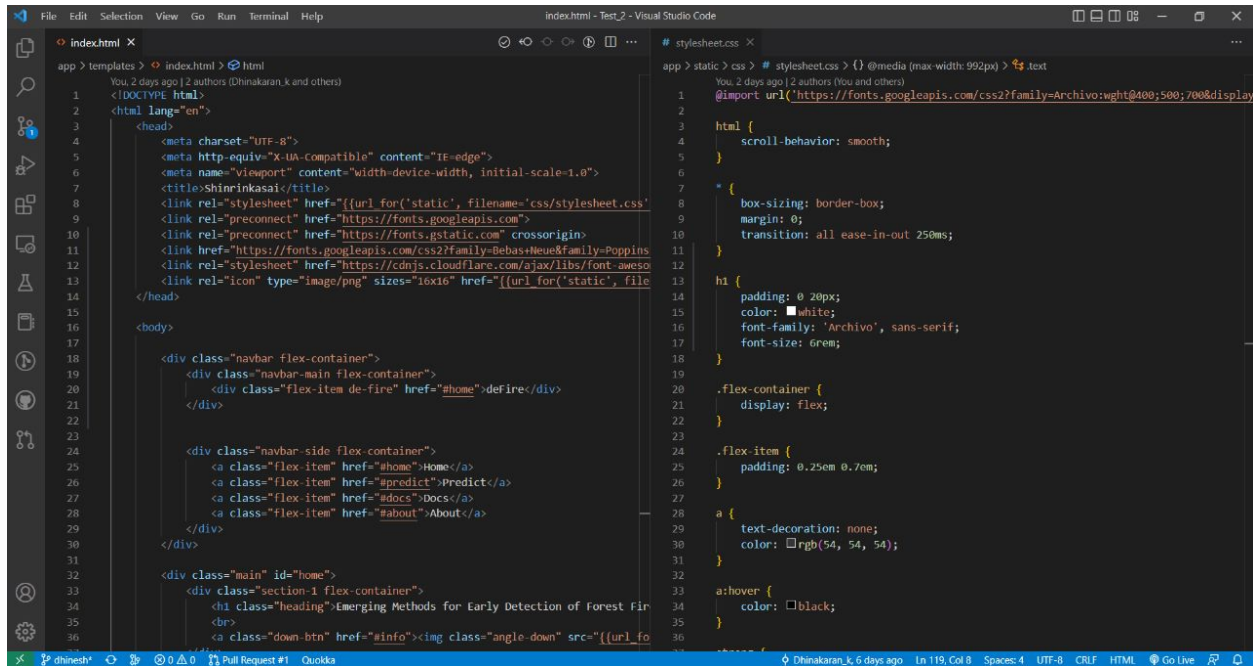
detect.py > Jupyter > predict
Dhinakaran_K, 6 days ago | 1 author (Dhinakaran_K)
1 import cv2
2 import os
3 import numpy as np
4 from tensorflow.keras.preprocessing import image
5 from tensorflow.keras.models import load_model
6 from Alert import send_alert, is_valid_number
7 import sys
8 import warnings
9 warnings.filterwarnings("ignore")
10 from dotenv import load_dotenv
11 load_dotenv('.env')
12
13 # Loading the model
14 model = load_model(os.getenv('MODEL_PATH'))
15
16 # Predict the image with Path as Parameters
17 def predict(img_path):
18     img = image.load_img(img_path, target_size=(64,64))
19     x = image.img_to_array(img)
20     x = np.expand_dims(x, axis=0)
21     pred = model.predict(x).flatten()[0]
22     return pred
23
24 # Main detect Function
25 def detect(path=0, to=bool=False):
26     ...
27     This is function is for Detecting The Forest Fire In the live Webcam
28     Model is loaded and path can be changed with making changes in .env file
29     this function will awake the alarm if it finds the Fire
30
31     accepts a path parameter which you can use to load local file for sample prediction
32     Default = Liveweb cam
33     accepts a to parameter which you can use to load To Number for Alert
34
35
36

```

Figure – 7.1: Python code

7.2 Feature 2

HTML is the basic markup language used to build the structure of the website. CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once.



The screenshot shows a Visual Studio Code editor with two files open: `index.html` and `stylesheet.css`. The `index.html` file contains the following code:

```

1 <!DOCTYPE html>
2 <html lang="en">
3
4   <head>
5     <meta charset="UTF-8">
6     <meta http-equiv="X-UA-Compatible" content="IE=edge">
7     <meta name="viewport" content="width=device-width, initial-scale=1.0">
8     <title>Shinrinkasai</title>
9     <link rel="stylesheet" href="{url for('static', filename='css/stylesheet.css')}">
10    <link rel="preconnect" href="https://fonts.googleapis.com">
11    <link href="https://fonts.googleapis.com/css2?family=Bebas+Neue&family=Poppins" rel="stylesheet">
12    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css" rel="stylesheet">
13    <link rel="icon" type="image/png" sizes="16x16" href="{url for('static', file
14  </head>
15
16  <body>
17
18    <div class="navbar flex-container">
19      <div class="navbar-main flex-container">
20        <div class="flex-item de-fire" href="#home">Home</div>
21      </div>
22
23      <div class="navbar-side flex-container">
24        <a class="flex-item" href="#home">Home</a>
25        <a class="flex-item" href="#predict">Predict</a>
26        <a class="flex-item" href="#docs">Docs</a>
27        <a class="flex-item" href="#about">About</a>
28      </div>
29    </div>
30
31    <div class="main" id="home">
32      <div class="section-1 flex-container">
33        <div class="heading">Emerging Methods for Early Detection of Forest Fir
34        <div class="down-btn" href="#info"><img class="angle-down" src="{url fo
35      </div>
36    </div>

```

The `stylesheet.css` file contains the following code:

```

1 @media (max-width: 992px) {
2   @import url('https://fonts.googleapis.com/css2?family=Archivo:wght@400;500;700&display=swap');
3
4   html {
5     scroll-behavior: smooth;
6   }
7
8   * {
9     box-sizing: border-box;
10    margin: 0;
11    transition: all ease-in-out 250ms;
12  }
13
14  h1 {
15    padding: 0 20px;
16    color: white;
17    font-family: 'Archivo', sans-serif;
18    font-size: 6rem;
19  }
20
21  .flex-container {
22    display: flex;
23  }
24
25  .flex-item {
26    padding: 0.25em 0.7em;
27  }
28
29  a {
30    text-decoration: none;
31    color: rgb(54, 54, 54);
32  }
33
34  a:hover {
35    color: black;
36  }

```

Figure – 7.2: HTML and CSS code

CHAPTER 8

TESTING

8.1 Test Cases

A test case might be created as an automated script to verify the functionality per the original acceptance criteria. After doing manual exploratory testing, QA testers might suggest other functionality be added to the application as well as updated test cases be incorporated in the automated test suite.

Table 8.1. Test Case

Test case ID	Feature Type	Component	Test Scenario
SMS Notification TC	Twilio SMS Notification	Python	Verify if user is able to receive SMS when forest fire is detected in the video processed by the model
Deployment TC	Website Deployment	Azure	Website built using HTML and designed using CSS is deployed using Microsoft Azure
Frontend & Backend TC	Website Functionality	Home page (Client)	Verify if front-end and back-end are well connected and the results are as expected.

Table.8.2. Test Report

Steps To Execute	Test Data	Expected Result	Status	Executed By
1. Execute script with intended video file to check. 2. Check if the results are expected. 3. Note the results.	detect.py	User should receive SMS Notification	Pass	Revathy V
1. Deploy repository on Azure 2. Check if the website is live. 3. Note the results.	forest-fire.azure-websites.net	Website should be live in the given URL	Pass	Dhinakaran K
1. Upload an image to the website to check the model's working 2. Check if the results are as expected. 3. Note the results.	forest-fire.azure-websites.net	We should get expected detection results for the uploaded image	Pass	Santhosh S Shyam V

8.2 User Acceptance Testing

Test Scenario:

Predict the Output.

Description: To predict the output for the given input video or image.

Test Step:

Model:

1. Choose Video file or use default video or use webcam input.
2. Execute the program.
3. If fire detected user receives SMS alert and console also displays and sounds an alert.

Website:

1. Choose Image file as input.
2. Click upload.
3. Website shows the result – ‘Positive’ or ‘Negative’.

Expected Result: Should display the exact prediction.

Actual Result: As Expected.

Status: PASS.

Table – 8.2: Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	-	-	7
Client Application	9	-	-	9
Security	1	-	-	1
Exception Reporting	3	-	-	3
Final Report Output	3	-	-	3
Version Control	4	1	-	3

CHAPTER 9

RESULTS

9.1 Performance Metrics

The Depicting Accuracy graph shows the Training and Validation Accuracy along the Epochs (x-axis) and Accuracy (y-axis).

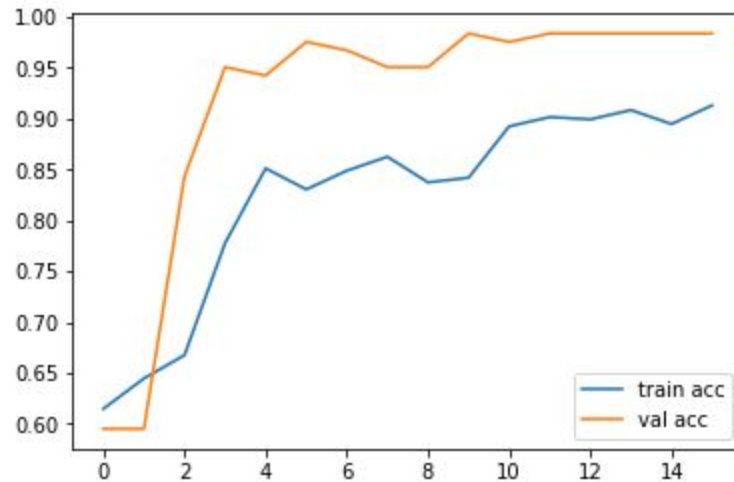


Figure - 9.1.1: Plot Depicting Accuracy

The Depicting Loss graph shows the Training and Validation Loss along the Epochs (x-axis) and Loss (y-axis).

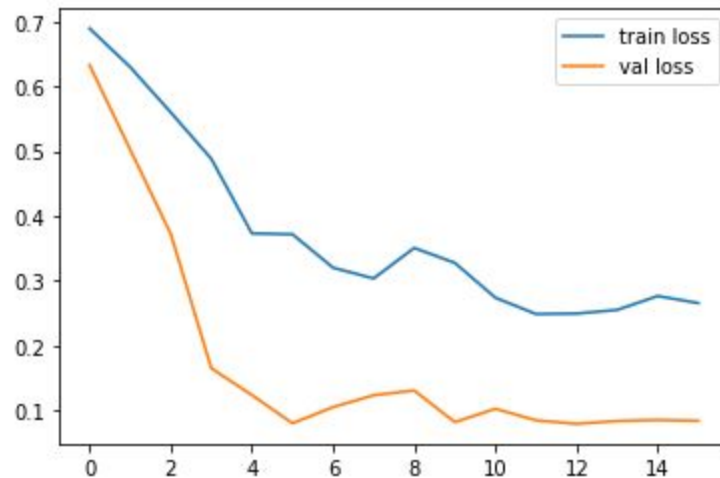


Figure - 9.1.2: Plot Depicting Loss

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- This project helps forest officers and fire fighters to respond quickly to the forest fire so that they can handle it in its earlier phases.
- This project can be scaled to a large area easily given that there are a few basic requirements fulfilled.
- The detection accuracy obtained in this project is much better as compared to the existing methodologies used. It can produce significant results and with even more live data available to train, the model can be improved much more decreasing false results.

DISADVANTAGES

- The current version of the project cannot handle large amount of data for processing i.e., detecting the forest fire.
- There is no clear graphical user interface to handle and organize all video input efficiently.

CHAPTER 11

CONCLUSION

The project mainly helps forest officers and fire fighters to prevent forest fires and stop the forest fires from spreading. It also helps police officers and environmentalists to support in the rescue process. Forest fires pose a great threat to the environment, they decrease the quality of forests, endanger many species of flora and fauna resulting in depletion of natural resources and loss of human lives.

The current response time for handling the forest fires is too long. The delay in response can cause a fatal accident and it also increases the probability of the fire spreading wider. So, this project detects forest fire using Deep Learning and immediately alerts responsible people with SMS alert. It aims to decrease the response time to limit the damage by fighting the fire in its weak beginning phase.

CHAPTER 12

FUTURE SCOPE

The current version of this project sends SMS alert to a single registered number using Twilio API. It also has a lower video processing capacity – to both capture and to detect forest fires. Some other additional features that are planned to be incorporated with this existing product are listed below:

- User can fetch multiple live cam input using a more powerful and robust processing system.
- User can use a latitude and longitude-based camera system to survey the forest area completely while scanning for animal movement to make sure of their presence in the region.
- User can also use UAV or drones in our response team to assist the fire fighters while also capturing real-time data.
- User can also create a more enhanced dashboard with more than binary response, we can include live temperature and natural gas levels (caused by decomposing material) and a quick response system to improve efficiency and decrease response time.

CHAPTER 13

APPENDIX

13.1 Source Code

MODEL

Model Building

```
model = Sequential()
#convolution and Pooling layer 1
model.add(Conv2D(filters=48,kernel_size=3,activation='relu',input_shape=(64,64,3)))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.2))
#convolution and Pooling layer 2
model.add(Conv2D(filters=32,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.2))
#Flattening the images
model.add(Flatten())
#Fully Connected layers
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1,activation='sigmoid'))
# Adding call to Avoid Overfitting
early_stop = EarlyStopping(monitor="val_accuracy",
                           min_delta=0.003,
                           patience=6,
                           verbose=1,
                           mode='auto',
```

```
        restore_best_weights=True)
lr = ReduceLROnPlateau(monitor="val_accuracy",
                        factor=0.2,
                        patience=3,
                        verbose=1,
                        mode="auto",
                        min_delta=0.003,
                        cooldown=1)
callback = [early_stop,lr]
# Fit The Model
result = model.fit(train_data,
                   epochs=30,
                   callbacks=callback,
                   validation_data=test_data)
```

Detect.Py

```
import cv2
import os
import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
from Alert import send_alert, is_valid_number
import sys
import warnings
warnings.filterwarnings("ignore")
from dotenv import load_dotenv
load_dotenv('.env')
# Loading the model
model = load_model(os.getenv('MODEL_PATH'))
```

Predict the image with Path as Parameters

```
def predict(img_path):  
    img = image.load_img(img_path,target_size=(64,64))  
    x = image.img_to_array(img)  
    x = np.expand_dims(x,axis=0)  
    pred = model.predict(x).flatten()[0]  
    return pred
```

Main detect Function

```
def detect(path=0,to:bool=False):  
    """  
    This is function is for Detecting The Forest Fire In the live Webcam  
    Model is Loaded and path can be changed with making changes in .env file  
    this function will awake the alarm if it finds the Fire  
    accepts a path parameter which you can use to load local file for sample prediction  
    Default = Liveweb cam  
    accepts a to parameter which you can use to load To Number for Alert  
    Default = which is prebuild Number  
    will write the final image predicted in Name Of : Files/Image.jpg  
    and write the prediction in a file named of : Files/prediction.txt  
    Press 'q' to stop the Program  
    """  
    video = cv2.VideoCapture(path)  
    if not video.isOpened():  
        print("Could not open webcam")  
        exit()  
    while True:  
        success,frame = video.read()
```

```
cv2.imwrite('Files/Image.jpg',frame)
pred = predict('Files/Image.jpg')
with open('Files/prediction.txt','w') as f:
    f.write(f"{str(pred)}\n1")
if not pred:
    cv2.imshow('Video',frame)
    print('No Issues')
else:
    cv2.putText(frame,"Fire Detected!!!",(50,70),fontFace=
cv2.FONT_HERSHEY_SIMPLEX,thickness=3,fontScale=2,color=(0,0,255))
    cv2.imshow('Video',frame)
    print('There is Fire!!')
    if to:
        send_alert(To=to)
    else:
        send_alert()
    if cv2.waitKey(1)&0xFF == ord('q'):
        break
video.release()
cv2.destroyAllWindows()
with open('Files/prediction.txt','w') as f:
    f.write(f"0\n0")
print("Program Closing...")
if __name__=="__main__":
    path = 0
    to = False
    # For To Number
    if '--to' in sys.argv:
```

```
to = sys.argv[(sys.argv.index('--to')) + 1]
if not is_valid_number(to):
    print('Enter a valid Twilio Verified Number')
    exit(0)
# For test video path
if '--path' in sys.argv:
    path = sys.argv[(sys.argv.index('--path')) + 1]
# For pre-build Test vidwo
if 'test' in sys.argv:
    path = 'Files/Test_Video_2.mp4'
# Primary Livecam Start
detect(path,to)
```

13.2 GitHub and Project Demo Link

GitHub - <https://github.com/IBM-EPBL/IBM-Project-39552-1660457369>

Project Demo Link - <https://www.youtube.com/embed/GFqn8cOcoy0>

Website Link - <https://forest-fire.azurewebsites.net/>

CHAPTER 14

REFERENCES

- [1] Diyana Kinaneva, Georgi Hristov, Jordan Raychev and Plamen Zahariev (2019) ‘Early Forest Fire Detection Using Drones and Artificial Intelligence’, pp.1060-1065.
- [2] A. Aksamovic, M. Hebibovic, and D. Boskovic (2017) ‘Forest fire early detection system design utilising the WSN simulator ', *XXVI International Conference on Information, Communication and Automation Technologies (ICAT)*, pp.1-5.
- [3] A. Sai Chand, K. Sai Bhargavi, R Sai Kiran, M.K. Kaushik, D.Raghavi Prashanthi and S. Siva Kumar (2018) ‘SAMRAKSHA : Developing a real-time and automatic early warning system for forest fire’, *IEEE Conference*.
- [4] Benamar Kadri, Benamar Bouyeddou and Djillali Moussaoui (2018) ‘Early Fire Detection System Using Wireless Sensor Networks’, *International Conference on Applied Smart Systems*.
- [5] Jiye Qian, Jin Fu, Jide Qian, Weibin Yang, Ke Wang and Pan Cao (2018) ‘Automatic Early Forest Fire Detection Based on Gaussian Mixture Model’, *18th IEEE International Conference on Communication Technology*, pp.1192-1196.
- [6] V. C. Moulianitis, G. Thanellas, N. Xanthopoulos and N. A. Aspragathos (2018) ‘Evaluation of UAV based schemes for forest fire monitoring’, *27th International Conference on Robotics in Alpe-Adria-Danube Region*.
- [7] Georgi Hristov, Jordan Raychev, Diyana Kinaneva and Plamen Zahariev (2018) ‘Emerging methods for early detection of forest fires using unmanned aerial vehicles and LoRaWAN sensor networks’.

- [8] Q. Zhang, G. Lin, Y. Zhang, G. Xu, and J. Wang (2018) ‘Wildland forest fire smoke detection based on faster R-CNN using synthetic smoke images’ *Procedia Engineering*, vol. 211, pp. 441–446.
- [9] Taanya Gupta, Hengyue Liu and Bir Bhanu (2021) ‘Early Wildfire Smoke Detection in Videos’, *25th International Conference on Pattern Recognition*, pp.8523-8530.
- [10]J. Shi, W. Wang, Y. Gao and N. Yu (2020) ‘Optimal placement and intelligent smoke detection algorithm for wildfire-monitoring cameras’, *IEEE Access*, vol. 8, pp. 72326–72339.