**Project title:** Emerging Methods for Early Detection of Forest Fires

**TEAM ID**: PNT2022TMID21021

**TEAM MEMBERS**: YASWANTH KUMAR YENDLURI

SARAN R

JEREMIN S H

SURESHKUMAR S

VARADHARAJ K

**Introduction:**

* 1. **Project overview:**

Fire can make major hazards in this hectic world. All buildings and vehicles used in public transportation have fire prevention and fire protection systems due to the accelerated number in the fire incidents. Also, many of the firms conduct a mock fire drill in every occurrence of months to protect their employees from the fire. This would help them to understand what to do or what not to do when a fire situation happens. Forests are one of the main factors in balancing the ecology. It is very harmful when a fire occurs in a forest. But most of the time, the detection of forest fire happens when it spread over a wide region. Sometimes, it could not be possible to stop the fire. As a result, the damage of the environment is higher than predictable. The emission of large amount of carbon dioxide (CO2) from the forest fire damages the environment. As well as it would lead to complete disappearance of rare species in the world (Alkhatib, 2014). Also, it can make an impact on the weather, and this make major issues like earthquakes, heavy rains, floods and so on.

A research study shows an automatic fire detection can be divided into three groups: aerial, ground and borne detection. The ground-based systems use several staring black and white video cameras are used in fire detection which detect the smoke and compares it with the natural smoke. The main benefit of using this system is high temporal resolution and spatial resolution. So that, the detection is easier (Eric den breejen, 1998). But these mechanisms still have some drawbacks in detecting the early stage of the fire. So that, it is highly important to introduce a system to detect the fire early as possible.

* 1. *Purpose:*

## we propose a novel system for detecting fire using Convolutional Neural Networks (CNN). Detection of fire can be extremely difficult using existing methods of smoke sensors installed in the buildings. They are slow and cost inefficient due to their primitive design and technology. This paper critically analyzes the scope of Artificial intelligence for detection and sending alerts with video from CCTV footages. This project uses self-built dataset containing video frames with fire. The data is then preprocessed and use the CNN to build a machine learning model. The test set of the dataset is given as input for validating the algorithm and experiments are noted. The

**project focus on building cost efficient and highly accurate machine that can be used in almost any use case of fire detection.**

***Keywords:Fire detection, Convolutional neural networks, Machine learning, CCTV, Object detection.***

1. LITERATURE SURVEY:
   1. Existing Problem:

The existing system for detecting fire are smoke alarms and heat alarms. The main disadvantage of the smoke sensor alarm and heat sensor alarms are that just one module is not enough to monitor all the potential fire prone places. The only way to prevent a fire is to be cautious al the time. Even if they are installed in every nook and corner, it just is not sufficient for an efficient output consistently. As the number of smoke sensor requirement increase the cost will also increase to its multiple. The proposed system can produce consistent and highly accurate alerts within seconds of accident of the fire. It reduces cost because only one software is enough to power the entire network of surveillance. Research is active on this field by data scientists and machine learning researchers. The real challenge is to minimize the error in detection of fire and sending alerts at the right time.

* 1. References

1. Saponara, S., Elhanashi, A. & Gagliardi, A. Real-time video fire/smoke detection based on CNN in antifire surveillance systems. J Real-Time Image Proc 18, 889–900 (2021).
2. Qingjie Zhang, Jiaolong Xu, Liang Xu and Haifeng Guo, Deep Convolutional Neural Networks for Forest Fire Detection. IFMEITA 2016.
3. Pragati, S. S. (2019-2020). International Journal Of Advance Scientific Research. Forest Fire Detection Using Machine Learning,

[

4] A Arul, R. S. (2021, May). Fire Detection System Using Machine Learning. Retrieved from ResearchGate:

[https://www.researchgate.net/publication/351926970\_Fire\_Detection\_System\_Using\_Machin](https://www.researchgate.net/publication/351926970_Fire_Detection_System_Using_Machine_Learning) [e\_Learning.](https://www.researchgate.net/publication/351926970_Fire_Detection_System_Using_Machine_Learning)

1. Yuanbin Wang, L. D. (2019). Journal of algorithms and Computational technology. Forest fire image recognition based on convolutional neural network, 1.
2. Kim, Byoungjun, and Joonwhoan Lee. 2019. "A Video-Based Fire Detection Using Deep Learning Models" *Applied Sciences* 9, no. 14: 2862
3. Xu, R.; Lin, H.; Lu, K.; Cao, L.; Liu, Y. A Forest Fire Detection System Based on Ensemble Learning. *Forests* **2021**, *12*, 217. https://doi.org/10.3390/f12020217
4. BoWFire Dataset. Available online: https://bitbucket.org/gbdi/bowfire-dataset/downloads/ (accessed on 1 January 2021). 29.
5. FD-Dataset. Available online: <http://www.nnmtl.cn/EFDNet/> (accessed on 1 January 2021).
6. ForestryImages. Available online: https:/[/www.for](http://www.forestryimages.org/browse/subthumb.cfm?sub=740)e[stryimages.org/browse/subthumb.cfm?sub=740](http://www.forestryimages.org/browse/subthumb.cfm?sub=740) (accessed on 1 January 2021). 31. [10].
7. VisiFire. Available online: <http://signal.ee.bilkent.edu.tr/VisiFire/> (accessed on 1 January 2021).

**Problem Statement:**

In earlier times fires were detected with the help of watching towers or using satellite images.

* Satellites collect images and send it to the monitoring authority which will decide by seeing images that it is a fire or not.
* But this approach was very slow as the fire may have spread in the large areas and caused so much damage before the rescue team came.
* In the watching tower method, there was a man always standing on the tower who would monitor the area and inform if there was fire.
* This method was also slow because before the man got to know about the fire it may have spread in the inner parts of forest, also it always requires a man who must be present there.
* Since, we know that some areas, especially forest areas are large so it is practically impossible to put a man in every part of forest from where they can monitor the forest area.
* So, both these approaches of watching towers and satellite images failed to detect fire as early as possible to reduce the damage done by fire Problems in fire detection:
* There were mainly two problems in fire detection as discussed:
* (a). Judging criteria for the fire: Edge is set, on the off chance that the worth is more noteworthy than edge, it is a fire, else not.
* So, this problem was removed by using machine learning techniques by many researchers.
* (b). Connection of nodes: Traditional systems used cables to connect alarm with the detectors.
* Cable was mainly of copper. But copper wire may be costly or it can suffer from fault in the mid-way.
* So, this problem was removed using wireless sensor networks.
* So, with the advancement in technology researchers find an efficient method to detect forest fire with the help of Wireless Sensor Network.
* Fire can be identified by conveying sensor hubs in timberland regions by which they illuminate about fire.
* Conveying sensor hubs in the timberland regions means placing sensors in every part of the forest and mostly in the prone areas where risk of 9 catching fire is more. With the use of wireless sensor networks, now it is easy to detect the fire in large areas as soon as possible.

**Ideation & Proposed Solution:**

# Empathy Map Canvas:

**Ideation & Brainstorming: Proposed solution:**

**Proposed Solution Template:**

Project team shall fill the following information in proposed solution template.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | Detecting forest fires in an early stage to avoid massive damage. |
| 2. | Idea / Solution description | Identifying huge forest fires in real-time utilising AI algorithms with camera and satellite footage.  The systems then notify dispatchers and local  authorities about the new ignition |
| 3. | Novelty / Uniqueness | Convolutional Neural Network system allows us to deliver information more quickly and accurately. It is possible to deploy a comprehensive coverage, which is nearly  impossible. |
| 4. | Social Impact / Customer Satisfaction | Monitoring of the potential danger regions and early identification of fire can greatly minimise the response time, as well as potential damage and firefighting expenses, while also saving  many lives. |
| 5. | Business Model (Revenue Model) | Subscription Model |
| 6. | Scalability of the Solution | Despite the physical distance between resources and users, its regionally scalable  system maintains its usability and utility; |

## REQUIREMENT ANALYSIS :

* 1. **FUNCTIONAL REQUIREMENT:**

SYSTEM 13

HARD DISK 500

RAM :4gb DDR2

BOARD :LG 104 KEYS KEYBOARD

MOUSE ;LOGITECH

MONITOR ; 15INCH TFT COLOR MONITOR

## NON-FUNCTIONAL REQUIREMENT :

Software Specifications Google Colaboratory — Colaboratory is a free Jupyter notebook environment provided by Google where one can use free GPUs and TPUs which requires no setup and runs entirely in the cloud. The Jupyter Notebook is an open-source web application which allows to create and share documents that contain live code, equations, visualizations and narrative text[11]. A notebook is a list of cells. Cells contain either explanatory text or executable code and its output. With Colaboratory one can write and execute code, save and share their analyses, and access powerful computing resources.

## PROJECT DESIGN:

* 1. **DATA FLOW DIAGRAM Data Flow Diagrams:**

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

SIMPLIFIED DATA:

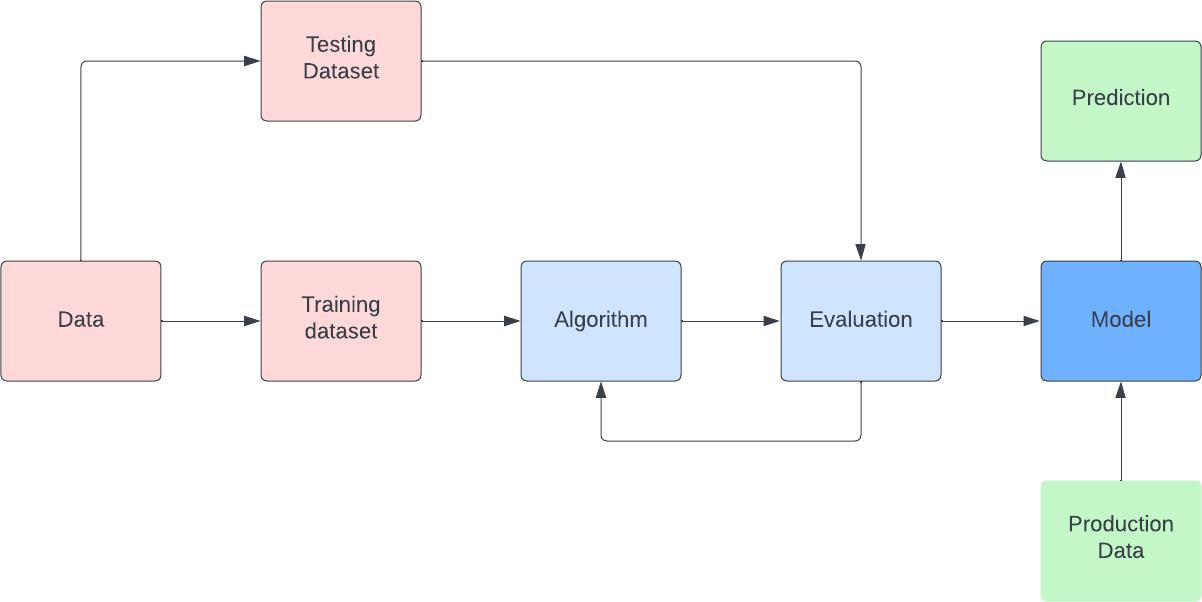
1. COLLLECT DATA

2. EVALUATE DATASET

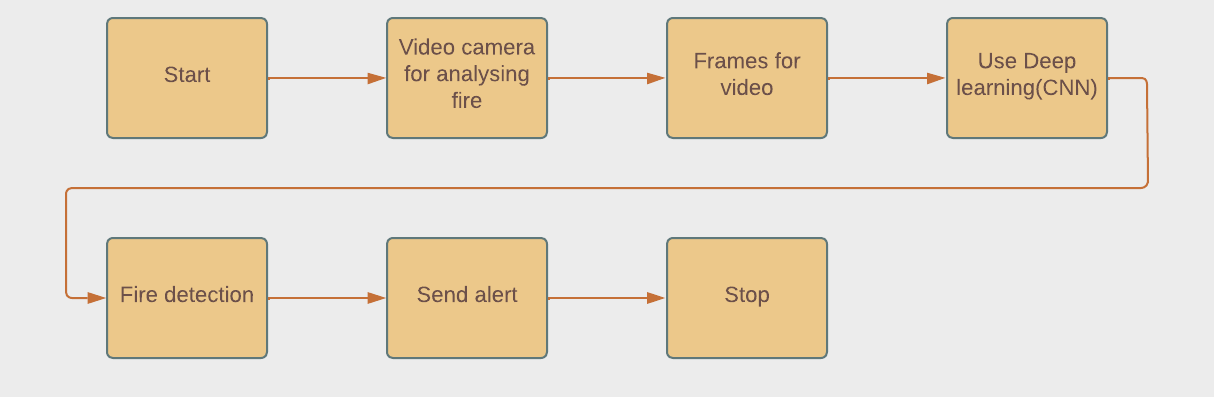
3. IMPLEMENT ALGORITHMS

4. EVALUATE THE ACCURACY FOR EACH ALGORITHMS

5. DISPLAY THE REUSLTS



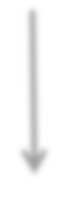
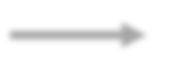
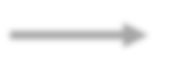
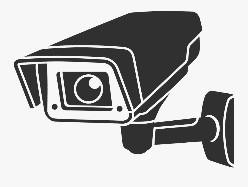
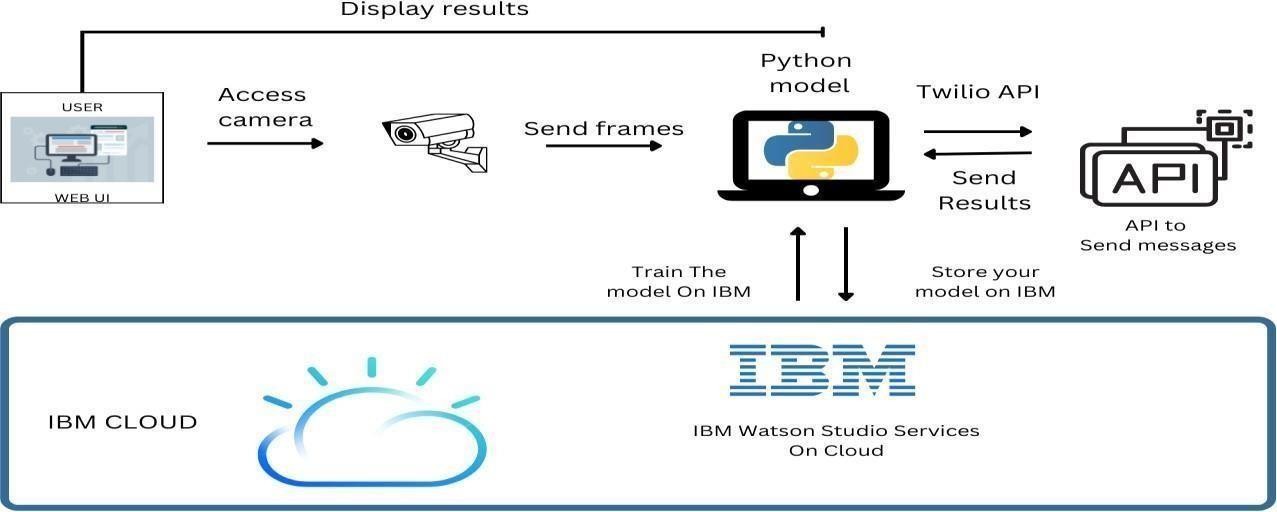
**Industry Standard DFD:**



**User Stories:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User type** | **Functional Requirement** | **User story number** | **User Story/ Task** | | **Acceptance criteria** | | **Priority** | **Release** |
| Environmentalist | Collect the data | USN -1 | Environmentalists help the public make informed decisions about the use of limited natural resources. They do research, produce reports, write articles, lecture, issue press releases, lobby congress, fundraise, and campaign. The daily routine depends  on the specialty. | | It is necessary to collect the right data else the prediction may become wrong. | | High | Sprint – 1 |
|  |  | USN - 2 | Identify algorithms that can be used for prediction. | | To collect the algorithm to identify the accuracy level of each algorithm. | | Medium | Sprint - 2 |
|  | Implement algorithm | USN - 3 | Identify the accuracy of all algorithms that are being used. | the | Accuracy of each algorithm is calculated so that it is  easy to obtain the most accurate output. | | High | Sprint – 2 |
|  |  | USN - 4 | Evaluate the Dataset. | | Data is evaluated before processing. | | Medium | Sprint – 1 |
|  | Evaluate accuracy of algorithm | USN - 5 | Identify accuracy, precision recall of each algorithm. | and | These values important  obtaining the output. | are for right | High | Sprint – 3 |
|  | Display results | USN - 6 | Outputs from each algorithm obtained. | are | It is highly used to predict the effect and to take precautionary measures | | High | Sprint - 4 |

**TECHNOLOGY ARCHITECTURE**



**Table-1:**

**Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with application e.g. sensor based system,application based system | Python/HTML ,CSS , Java script and react Js |
| 2. | Input | Video Feed | Web Camera/Video on a site |
| 3. | Application Logic-1 | Logic for a process in the application e.g: registration process. | Python |
| 4. | Application Logic-2 | Logic for a process in the application e.g registration process successful then go to login process | Python |
| 5. | Dataset | Using Test set and Train set, train the model | Data set from Cloud Storage, Database |
| 6. | Cloud Database | Database Service on Cloud | IBM DB2, IBM Cloudant etc. |
| 7. | Infrastructure (Server / Cloud), API | Application Deployment on Local System /Cloud Local,Cloud Server Configuration. | Local, Cloud Foundry, Kubernetes, etc. |

**Table-2:**

**Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | open-source frameworks used. | Technology of Opensource framework |
| 2. | Security Implementations | Mandatory Access Control (MAC) and Preventative Security Control is used | e g. SHA-256, Encryptions, IAM Controls, OWASP etc. |
| 3. | Scalable Architecture | scalability of architecture e g: early alarm in real time when the forest occurs. | Technology used |
| 4. | Availability | availability of application (e g. customer service and support to the mobile) | Technology used |
| 5. | Performance | Enhance the performance by using IBM CDN | IBM Content Delivery Network |

Solution Requirements (Functional & Non-functional)

# Functional Requirements:

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | **Classification** | There are four classifications of fire cause: accidental, natural, incendiary, and undetermined |
| FR-2 | **Tagging** | It is a classification task with a higher degree of precision. It helps to identify several objects within  an image. It tags and tracks animals and forest situations |
| FR-3 | **Localization** | The localisation of the node would be done using satellite communication to reduce coverage holes and ensure maximum range with the least latency. This node would communicate data to a monitoring station with its location and send alerts according to the sensed thresholds breached based on the novel  logic algorithm. |
| FR-4 | **Detection** | The system, using Moderate Resolution Imaging Spectro-Radiometer (MODIS), Advanced Very High Resolution Radiometer (AVHRR), and Spinning Enhanced Visible and Infra Red Imager (SEVIRI) data, provides near real-time integrated  information about both the fire presence and danger over the affected area |
| FR-5 | **Semantic Segmentation** | Semantic segmentation describes the process of associating each pixel of an image with a class  label. It includes Sentinel-1, Sentinel -2, Sentinel-3 and MODIS. |
| FR-6 | **Instance Segmentation** | In Instance Segmentation, bounding boxes are generated for each instance of multiple categories present along with the object segmentation masks.  It includes Hydrology, Rivers, Lakes and Audio weather conditions. |

# Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Many methods have been proposed to detect forest fires, such as camera-based systems, WSN-based systems, and machine learning application-based systems, with both positive and negative aspects and performance figures of detection. |
| NFR-2 | **Security** | As this process is designed with a minimum delay, the fire can be detected within the initial stage, and the responsible parties can take necessary actions in a shorter period, which will minimize the damage. This ensures security of well beings. |
| NFR-3 | **Reliability** | The system shall be supervised either electrically or with satellite or even by software- directed polling of field. The panel, detectors and modules shall preferably used. |
| NFR-4 | **Performance** | In the event of a fire, the primary objective of using drones is to gather situational awareness, which can be used to direct the efforts of the firefighters in locating and controlling hot spots. Just like urban fires, forest fires to require monitoring so that firefighters know what they are dealing with. Model will achieve high accuracy |
| NFR-5 | **Availability** | By making field testing ,Threshold ratio analysis  it ensures minimum up time and performance |
| NFR-6 | **Scalability** | A widely used measure of fire intensity is fireline intensity, which is the rate of heat transfer per unit length of the fire line (measured in kW m−1) and represents the radiant energy release in the flaming front, |

## PROJECT PLANNING &ESTIMATION

* 1. **SPRINT PLANNING & ESTIMATION &**

## SPRINT DELIVERY SCHEDULE

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement** | **User story number** | **User Story/ Task** | **Acceptance criteria** | **Priority** | **Team Members** |
| Sprint – 1 | Collect the data | USN -1 | Environmentalists help the public make informed decisions about the use of limited natural resources. They do research, produce reports, write articles, lecture, issue press releases, lobby congress, fundraise, and campaign. The daily routine depends on the specialty. | It is necessary to collect the right data else the prediction may become wrong. | High | * Saran * Varadharaj * Yaswanth * Jeremin * Suresh |
| Sprint - 2 | Image preprocessing | USN - 2 | Identify algorithms that can be used for prediction. | To collect the algorithm to identify the accuracy level of  each algorithm. | Medium |
| Sprint – 2 | Implement algorithm | USN - 3 | Identify the accuracy of all the algorithms that are being used. | Accuracy of each algorithm is calculated so that it is easy to obtain the most accurate output. | High |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint – 1 | Reviewing the model | USN - 4 | Evaluate the Dataset. | Data is evaluated before processing. | Medium | * Saran * Varadharaj * Yaswanth * Jeremin * Suresh |
| Sprint – 3 | Evaluate accuracy of algorithm | USN - 5 | Identify accuracy, precision and recall of each algorithm. | These values are important for obtaining the right  output. | High |
| Sprint - 4 | Display results | USN - 6 | Outputs from each algorithm are obtained. | It is highly used to predict the effect and to take precautionary measures | High |

**Project Tracker, Velocity & Burndown Chart:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date (Actual)** |
| Sprint-1 | 20 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 8 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 7 | 08 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 8 | 15 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 7 | 20 Nov 2022 |

## Project Tracker, Velocity & Burndown Chart: (4 Marks) Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

AV = Sprint duration /velocity = 7/10 =0.7

## CODING & SOLUTIONING

* 1. **FEATURE**

**"""Applying ImageDataGenerator functionality To Testset And Trainset.ipynb**

**Automatically generated by Colaboratory.**

**Original file is located at**

**https://colab.research.google.com/drive/13PPU6URoKuCkNiC\_SciYaXrxAfuWTDMZ**

**Team ID: PNT2022TMID21021**

**Applying ImageDataGenerator functionality To Testset And Trainset**

**"""**

**import keras**

**from keras.preprocessing.image import ImageDataGenerator**

**#Defining the parameters/arguments for ImageDataGenerator class**

**train\_datagen = ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)**

**test\_datagen = ImageDataGenerator(rescale=1./255)**

**#Applying the ImageDataGenerator function to the trainset**

**x\_train=train\_datagen.flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/train\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')**

**#Applying ImageDataGenerator functionality to the testset**

**x\_test=test\_datagen.flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')**

**# -\*- coding: utf-8 -\*-**

**"""OpenCv\_for\_Video\_processing.ipynb**

**Automatically generated by Colaboratory.**

**Original file is located at**

**https://colab.research.google.com/drive/1Ae3ZYf\_2zodEVMaxnMFxoEu7AJRBmIvp**

**Team ID: PNT2022TMID21021**

**OpenCv for Video processing**

**"""**

**#importing required libraries**

**!pip install tensorflow**

**!pip install opencv-python**

**!pip install opencv-contrib-python**

**import tensorflow as tf**

**import numpy as np**

**from tensorflow import keras**

**import os**

**import cv2**

**from tensorflow.keras.preprocessing.image import ImageDataGenerator**

**from tensorflow.keras.preprocessing import image**

**train=ImageDataGenerator(rescale=1./255,**

**shear\_range=0.2,**

**rotation\_range=180,**

**zoom\_range=0.2,**

**horizontal\_flip=True)**

**train = ImageDataGenerator(rescale=1/255)**

**test = ImageDataGenerator(rescale=1/255)**

**train\_dataset = train.flow\_from\_directory("/content/drive/MyDrive/IBM/Dataset/Dataset/train\_set", target\_size=(128,128), batch\_size = 32,**

**class\_mode = 'binary' )**

**test\_dataset = test.flow\_from\_directory("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set", target\_size=(128,128), batch\_size = 32,**

**class\_mode = 'binary' )**

**test\_dataset.class\_indices**

**#to define linear initialisation import sequential**

**from keras.models import Sequential**

**#to add layer import Dense**

**from keras.layers import Dense**

**#to create convolution kernel import convolution2D**

**from keras.layers import Convolution2D**

**#import Maxpooling layer**

**from keras.layers import MaxPooling2D**

**#import flatten layer**

**from keras.layers import Flatten**

**import warnings**

**warnings.filterwarnings('ignore')**

**model = keras.Sequential()**

**model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))**

**model.add(MaxPooling2D(pool\_size=(2,2)))**

**model.add(Convolution2D(32,(3,3),activation='relu'))**

**model.add(MaxPooling2D(pool\_size=(2,2)))**

**model.add(Convolution2D(32,(3,3),activation='relu'))**

**model.add(MaxPooling2D(pool\_size=(2,2)))**

**model.add(Convolution2D(32,(3,3),activation='relu'))**

**model.add(MaxPooling2D(pool\_size=(2,2)))**

**model.add(Flatten())**

**model.add(Dense(150,activation='relu'))**

**model.add(Dense(1,activation='sigmoid'))**

**model.compile(loss = 'binary\_crossentropy', optimizer = "adam", metrics = ["accuracy"])**

**r = model.fit(train\_dataset, epochs = 5, validation\_data = test\_dataset)**

**pred = model.predict(test\_dataset)**

**pred = np.round(pred)**

**pred**

**print(len(pred))**

**model.save("/content/forest1.h5")**

**#import load\_model from keras.model**

**from keras.models import load\_model**

**#import image class from keras**

**import tensorflow as tf**

**from tensorflow.keras.preprocessing import image**

**#import numpy**

**import numpy as np**

**#import cv2**

**import cv2**

**model = load\_model("/content/forest1.h5")**

**import matplotlib.pyplot as plt**

**plt.plot(r.history['loss'],label='loss')**

**plt.plot(r.history['val\_loss'],label='val\_loss')**

**plt.legend()**

**plt.plot(r.history['accuracy'],label='acc')**

**plt.plot(r.history['val\_accuracy'],label='val\_acc')**

**plt.legend()**

**def predictImage(filename):**

**img1=image.load\_img(filename,target\_size=(128,128))**

**plt.imshow(img1)**

**y=image.img\_to\_array(img1)**

**x=np.expand\_dims(y,axis=0)**

**ctr=model.predict(x)**

**print(ctr)**

**if ctr==1:**

**plt.xlabel("Fire detected",fontsize=30)**

**elif ctr==0:**

**plt.xlabel("No fire detected",fontsize=30)**

**predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/2017\_10\_12\_09\_01\_56.jpg")**

**predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/FORESTFIRE (1).jpg")**

**predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/beautiful\_mountain\_forest\_wallpaper\_1920x1200.jpg")**

**predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/ring\_of\_fire\_bailey\_colorado\_rocky\_mountain\_forest\_wildfire\_picture\_id157384116.jpg")**

**pip install twilio**

**pip install playsound**

**pip install opencv-python**

**#import opencv librariy**

**import cv2**

**#import numpy**

**import numpy as np**

**#import image function from keras**

**from keras.preprocessing import image**

**#import load\_model from keras**

**from keras.models import load\_model**

**#import client from twilio API**

**from twilio.rest import Client**

**#imort playsound package**

**from playsound import playsound**

**#load the saved model**

**model = load\_model(r'/content/forest1.h5')**

**#define video**

**video = cv2.VideoCapture('/content/drive/MyDrive/forest fire.mp4')**

**#define the features**

**name = ['forest','with forest']**

**"""Predictions.ipynb**

**Automatically generated by Colaboratory.**

**Original file is located at**

**https://colab.research.google.com/drive/1PVTQugaCBW35ByIHoQO2bbNg8ZPyyl89**

**Team ID: PNT2022TMID21021**

**Predictions**

**"""**

**import keras**

**from keras.preprocessing.image import ImageDataGenerator**

**#Defining the parameters/arguments for ImageDataGenerator class**

**train\_datagen = ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)**

**test\_datagen = ImageDataGenerator(rescale=1./255)**

**#Applying the ImageDataGenerator function to the trainset**

**x\_train=train\_datagen.flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/train\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')**

**#Applying ImageDataGenerator functionality to the testset**

**x\_test=test\_datagen.flow\_from\_directory('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')**

**#import model building libraries**

**#To define Linear initialisation import Sequential**

**from keras.models import Sequential**

**#To add layers import Dense**

**from keras.layers import Dense**

**#To create Convolution kernel import Convolution2D**

**from keras.layers import Convolution2D**

**#import Maxpooling layer**

**from keras.layers import MaxPooling2D**

**#import flatten layer**

**from keras.layers import Flatten**

**import warnings**

**warnings.filterwarnings('ignore')**

**#initializing the model**

**model=Sequential()**

**#add convolutional layer**

**model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))**

**#add maxpooling layer**

**model.add(MaxPooling2D(pool\_size=(2,2)))**

**#add flatten layer**

**model.add(Flatten())**

**#add hidden layer**

**model.add(Dense(150,activation='relu'))**

**#add output layer**

**model.add(Dense(1,activation='sigmoid'))**

**#configure the learning process**

**model.compile(loss='binary\_crossentropy',optimizer="adam",metrics=["accuracy"])**

**#Training the model using 'fit' method**

**model.fit\_generator(x\_train,steps\_per\_epoch=14,epochs=10,validation\_data=x\_test,validation\_steps=4)**

**#saving the model**

**model.save("forest1.h5")**

**#import load\_model from keras.model**

**from keras.models import load\_model**

**#import image class from keras**

**from tensorflow.keras.preprocessing import image**

**#import numpy**

**import numpy as np**

**#import cv2**

**import cv2**

**#load the saved model**

**model = load\_model("forest1.h5")**

**img=image.load\_img('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/Bandipur\_fires\_2019.jpg')**

**x=image.img\_to\_array(img)**

**res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER\_CUBIC)**

**#expand the image shape**

**x=np.expand\_dims(res,axis=0)**

**pred=model.predict(x)**

"""Train\_image\_classification\_Model.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1YDGFKlN1CzQrYucsnSxHSIYiD36KOJCn

## Team ID: \*\*PNT2022TMID21021\*\*

"""

!pip install tensorflow

!pip install opencv-python

!pip install opencv-contrib-python

pip install opencv-python

import tensorflow as tf

import numpy as np

from tensorflow import keras

import os

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.preprocessing import image

train=ImageDataGenerator(rescale=1./255,

shear\_range=0.2,

rotation\_range=180,

zoom\_range=0.2,

horizontal\_flip=True)

train = ImageDataGenerator(rescale=1/255)

test = ImageDataGenerator(rescale=1/255)

train\_dataset = train.flow\_from\_directory("//content/drive/MyDrive/IBM/Dataset/Dataset/test\_set",

target\_size=(128,128),

batch\_size = 32,

class\_mode = 'binary' )

test\_dataset = test.flow\_from\_directory("///content/drive/MyDrive/IBM/Dataset/Dataset/train\_set",

target\_size=(128,128),

batch\_size = 32,

class\_mode = 'binary' )

test\_dataset.class\_indices

#to define linear initialisation import sequential

from keras.models import Sequential

#to add layer import Dense

from keras.layers import Dense

#to create convolution kernel import convolution2D

from keras.layers import Convolution2D

#import Maxpooling layer

from keras.layers import MaxPooling2D

#import flatten layer

from keras.layers import Flatten

import warnings

warnings.filterwarnings('ignore')

model = keras.Sequential()

model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Convolution2D(32,(3,3),activation='relu'))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Convolution2D(32,(3,3),activation='relu'))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Convolution2D(32,(3,3),activation='relu'))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Flatten())

model.add(Dense(150,activation='relu'))

model.add(Dense(1,activation='sigmoid'))

model.compile(loss = 'binary\_crossentropy',

optimizer = "adam",

metrics = ["accuracy"])

r = model.fit(train\_dataset, epochs = 5, validation\_data = test\_dataset)

predictions = model.predict(test\_dataset)

predictions = np.round(predictions)

print(len(predictions))

model.save("/content/forestmodel.h5")

#import load\_model from keras.model

from keras.models import load\_model

#import image class from keras

import tensorflow as tf

from tensorflow.keras.preprocessing import image

#import numpy

import numpy as np

#import cv2

import cv2

model = load\_model("/content/forestmodel.h5")

import matplotlib.pyplot as plt

plt.plot(r.history['loss'],label='loss')

plt.plot(r.history['val\_loss'],label='val\_loss')

plt.legend()

plt.plot(r.history['accuracy'],label='acc')

plt.plot(r.history['val\_accuracy'],label='val\_acc')

plt.legend()

def predictImage(filename):

img1=image.load\_img(filename,target\_size=(128,128))

plt.imshow(img1)

y=image.img\_to\_array(img1)

x=np.expand\_dims(y,axis=0)

val=model.predict(x)

print(val)

if val==0:

plt.xlabel("No fire Detected!!",fontsize=30)

elif val==1:

plt.xlabel("Fire Detected!!",fontsize=30)

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/19464620\_401.jpg")

predictImage("////content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/0.64133000\_1519374442\_forest\_deep.jpg")

pip install twilio

pip install playsound

pip install opencv-python

#import opencv librariy

import cv2

#import numpy

import numpy as np

#import image function from keras

from keras.preprocessing import image

#import load\_model from keras

from keras.models import load\_model

#import client from twilio API

from twilio.rest import Client

#imort playsound package

from playsound import playsound

#load the saved model

model = load\_model(r'/content/forestmodel.h5')

#define video

video = cv2.VideoCapture('//content/drive/MyDrive/IBM/forest video')

#define the features

name = ['forest','with forest']

video.isOpened()

from tensorflow.keras.preprocessing import image

from IPython.display import Audio

while(video.isOpened()):

success,frame=video.read()

cv2.imwrite("with fire/19464620\_401.jpg",frame)

img=image.load\_img("//content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/19464620\_401.jpg",target\_size=(128,128))

x=image.img\_to\_array(img)

x=np.expand\_dims(x,axis=0)

pred=model.predict(x)

p=pred[0]

print(pred)

cv2.putText(frame,"predicted class = ",(100,100),cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0,0,0), 1)

if pred[0]==1:

account\_sid = 'AC9f6fc46b088bc81bbeef4a802cd06864'

auth\_token = '29f7d63c9d6ce55afbecb683a85072c2'

client=Client(account\_sid,auth\_token)

message=client.messages \

.create(

body="Forest fire is detected ,stay alert",

from\_='+15095193634',

to='+919491291847')

print(message.sid)

print('Fire detected')

print('SMS sent')

wn=Audio('/content/tornado-siren.mp3',autoplay=True)

display(wn)

break

else:

print('No danger')

break

if cv2.waitKey(1) & 0xFF==ord('a'):

break

video.release()

cv2.destroyAllWindows()

**"""Final\_Code.ipynb**

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1x-IftWPDybimjmOL08Xrw-D1F6FAZ3VM

# \*\*Emerging Methods for Early Detection of Forest Fires\*\*

\*\*Team ID: PNT2022TMID21021\*\*

# \*\*Image Pre-Proecessing\*\*

\*\*1.Importing the ImageDataGenerator Library\*\*

"""

import numpy as np

import keras

from sklearn.model\_selection import train\_test\_split

from keras.models import Sequential, load\_model

from keras.preprocessing.image import ImageDataGenerator

from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard

from keras.callbacks import ReduceLROnPlateau

from keras.layers import Conv2D, Dropout, Dense, Flatten, MaxPooling2D, SeparableConv2D, Activation, BatchNormalization

import matplotlib.pyplot as plt

import time

import os

import tensorflow as tf

"""\*\*2.Define parameters for ImageDataGenerator Class\*\*"""

train\_datagen=ImageDataGenerator(rescale=1./255,

shear\_range=0.2,

rotation\_range=180,

zoom\_range=0.2,

horizontal\_flip=True)

test\_datagen=ImageDataGenerator(rescale=1./255)

"""\*\*3.Applying ImageDataGenerator Functionality to Trainset and Testset\*\*

a. for dataset

"""

x\_dataset =train\_datagen.flow\_from\_directory(r"/content/drive/MyDrive/IBM/Dataset/Dataset",target\_size = (128,128), class\_mode = "binary",batch\_size = 32)

"""b. for trainset"""

x\_train =train\_datagen.flow\_from\_directory(r"/content/drive/MyDrive/IBM/Dataset/Dataset/train\_set",target\_size = (128,128), class\_mode = "binary",batch\_size = 32)

"""c. for testset"""

x\_test =test\_datagen.flow\_from\_directory(r"/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set",target\_size = (128,128), class\_mode = "binary",batch\_size = 32)

x\_train.class\_indices

"""# \*\*Model Building\*\*

\*\*1. Importing Model Building libraries\*\*

"""

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

from tensorflow.keras.layers import Convolution2D

from tensorflow.keras.layers import MaxPooling2D

from tensorflow.keras.layers import Flatten

import warnings

warnings.filterwarnings('ignore')

"""\*\*2. Initializing the model\*\*"""

model=Sequential()

"""\*\*3. Adding CNN layers\*\*

\*a. Adding Convolution layers\*

"""

model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))

"""\*b. Adding pooling layer\*"""

model.add(MaxPooling2D(pool\_size=(2,2)))

"""\*c. Adding Flatten layer\*"""

model.add(Flatten())

model.summary()

"""\*\*4. Adding Dense layers\*\*

\*a. Adding hidden layers\*

"""

model.add(Dense(150,activation='relu'))

"""\*b. Adding Output layer\*"""

model.add(Dense(1,activation='sigmoid'))

"""\*\*5. Configuring the Learning Process\*\*

"""

model.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])

"""\*\*6. Training the model\*\*"""

#training using fit method

r=model.fit\_generator(x\_train,steps\_per\_epoch=14, epochs=10,validation\_data=x\_test, validation\_steps=2)

import matplotlib.pyplot as plt

plt.plot(r.history['loss'],label='loss')

plt.plot(r.history['val\_loss'],label='val\_loss')

plt.legend()

plt.plot(r.history['accuracy'],label='acc')

plt.plot(r.history['val\_accuracy'],label='val\_acc')

plt.legend()

"""\*\*7. Save the model\*\*"""

model.save("/content/forest\_model.h5")

"""\*\*8. Test the model\*\*"""

import numpy as np

from tensorflow.keras.models import load\_model

from tensorflow.keras.preprocessing import image

import cv2

model=load\_model('/content/forest\_model.h5')

img=image.load\_img('/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/Bandipur\_fires\_2019.jpg')

x=image.img\_to\_array(img)

res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER\_CUBIC)

#expand the image shape

x=np.expand\_dims(res,axis=0)

"""# \*\*Predictions\*\*"""

pred=model.predict(x)

pred

def predictImage(filename):

img1=image.load\_img(filename,target\_size=(128,128))

plt.imshow(img1)

y=image.img\_to\_array(img1)

x=np.expand\_dims(y,axis=0)

ctr=model.predict(x)

print(ctr)

if ctr==0:

plt.xlabel(" No Fire detected",fontsize=30)

elif ctr==1:

plt.xlabel("Fire detected",fontsize=30)

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/1170x500\_Ireland\_web.jpg")

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/01\_NeilBurnell\_Mystical\_photoverticall.jpg")

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/forest/2017\_10\_12\_09\_01\_56.jpg")

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/Forest\_fire\_MNRF\_esize\_IMG\_6743.jpg")

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/How\_to\_Protect\_Your\_Home\_From\_Forest\_Fire\_1024x588.jpg")

predictImage("/content/drive/MyDrive/IBM/Dataset/Dataset/test\_set/with fire/uttarakhand\_forest\_fire\_750x500.jpg")

"""# \*\*Sending Alert message\*\*"""

pip install twilio

pip install playsound

pip install opencv-python

#import opencv librariy

import cv2

#import numpy

import numpy as np

#import image function from keras

from keras.preprocessing import image

#import load\_model from keras

from keras.models import load\_model

#import client from twilio API

from twilio.rest import Client

#imort playsound package

from playsound import playsound

#load the saved model

model = load\_model(r'/content/forest\_model.h5')

#define video

video = cv2.VideoCapture('/content/drive/MyDrive/forest fire.mp4')

#define the features

name = ['forest','with forest']

from twilio.rest import Client

account\_sid = 'AC9f6fc46b088bc81bbeef4a802cd06864'

auth\_token = '29f7d63c9d6ce55afbecb683a85072c2'

client = Client(account\_sid, auth\_token)

message = client.messages.create(

messaging\_service\_sid='MG55c5ca2c99b3b2f04047a3b7fe504a56',

body='Forest fire is detected, stay alert!',

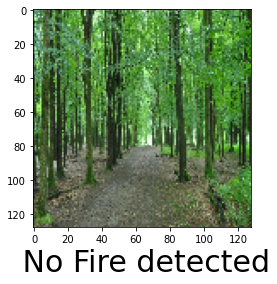
to='+918610271176'

)

print(message.sid)

print("Fire detected")

print("SMS Sent!")





RESULT:

* 1. **PERFORMANCE METRICES**

IX PERFORMANCE EVALUATION In this project, as of now, we have worked with two different machine learning models. We calculated the accuracy of these models. The comparison of these models is as follows: SVM Decision Tree Accuracy 0.62 0.99 Precision 0 0.54 0.98 Precision 1 0.76 1 Recall 0 0.78 0.99 Recall 1 0.51 0.98 Based on these observations after our experiment and analysis we can clearly compare the performance of the models to predict the chances of fire.

1. ADVANTAGE AND DISADVANTAGE:

Every year it seems like there’s another disastrous wildfire in the American West. In 2018, nearly 9 million acres were burned in the US alone. Uncontrolled fires often started accidentally by people, rampage and decimate forests. For most people, a forest fire is synonymous with disaster. But there are some kinds of forest fires that actually benefit the environment.

From forests to deserts, wildfires affect air quality, vegetation, human and animal habitats, and climate around the world. Fire managers and researchers are finding ways to use NASA data to battle fires and measure their effects. Burning fires produce both ashes, which falls to the ground like snow but can also get caught up in winds, and smoke, a mixture of gases and particulate matter. These get into the atmosphere and can travel long distances impacting air quality regionally. Wildfires are unplanned fires that start in forests or wildland areas. There are numerous post-fire impacts, including an increase in air pollution and less infiltration of precipitation, contributing to flooding hazards even long after the burn.

A controlled burn is a wildfire that people set intentionally for a specific purpose. Well- thought-out and well-managed controlled burns can be incredibly beneficial for forest management, in part because they can help stop an out-of-control wildfire. The technique is called backburning, and it involves setting a controlled fire in the path of the approaching wildfire. All the flammable material is burnt up and extinguished. When the wildfire approaches, there’s no more fuel left for it to keep going, and it dies out. Forest fire science entails understanding how a fire starts, what contributes to the fire and how the fire might impact future Earth processes. Understanding climatological changes are important to understand how these changes may contribute to fires in the future.

Controlled burns are also used to prevent forest fires. Even before human involvement, natural, low-intensity wildfires occurred every few years to burn up fuel, plant debris, and dead trees, making way for young, healthy trees and vegetation to thrive. That new growth in turn supports forest wildlife. Forest managers are now replicating this natural strategy when appropriate, starting manageable, slow-burning fires to make room for the new life that will help keep the forest healthy in the long term.

The same method is one of WWF’s strategies for maintaining grassland habitats in the Northern Great Plains. Working with partners such as the U.S. Fish and Wildlife Service, WWF has intentionally burned hundreds of acres of prairie land to revitalize these key habitats. The fire burns off tall, aggressive vegetation that isn’t as hospitable to wildlife, and makes room for new growth that attracts bison, birds, and prairie dogs.

This doesn’t mean all intentional wildfires are good. Many of the fires intentionally set for agriculture and land clearing are at best ill-advised, and at worst devastating. Slash and burn fires are set every day to destroy large sections of forests. Of course, these forests don’t just remove trees; they kill and displace wildlife, alter water cycles and soil fertility, and endanger the lives and livelihoods of local communities. They also can rage out of control. In 1997, fires set intentionally to clear forests in Indonesia escalated into one of the largest wildfires in recorded history. Hundreds of people died; millions of acres burned; already at-risk species like orangutans perished by the hundreds; and a smoke and ash haze hung over Southeast Asia for months, reducing visibility and causing acute health conditions.

When scientists think a fire could be the best solution for revitalizing wild areas, WWF brings the right experts to the table to study the situation and come up with a plan. All fire is risky. To minimize that risk as much as possible, controlled burns must be well-considered, well-planned, and ignited and maintained by trained professionals. Fire can be a tool for conservation, but only when used the right way.

Since 1990, “Wildland Fires” across Canada have consumed an average of 2.5 million hectares a year. These fires occur in forests, shrublands and grasslands. Some are uncontrolled wildfires started by lightning or human carelessness. A small number are prescribed fires set by authorized forest managers to mimic natural fire processes that renew and maintain healthy ecosystems. Wildland fires present a challenge for forest management because they have the potential to be at once harmful and beneficial. They can threaten communities and destroy vast amounts of timber resources, resulting in costly losses.

However, wildland fires are a natural part of the forest ecosystem and important in many parts of Canada for maintaining the health and diversity of the forest. In this way, prescribed fires offer a valuable resource management tool for enhancing ecological conditions and eliminating excessive fuel build-up.

Not all wildland fires should or can be controlled. Forest agencies work to harness the force of natural fire to take advantage of its ecological benefits while at the same time limiting its potential damage and costs. This makes fire control strategies a vital component of forest management and emergency management in Canada. Understanding the complex phenomenon of wildland fire begins with understanding the basic physical aspects of fire and the ecological role of fire in forests and other wildland areas. Increasingly accurate assessments of the fire situation across Canada are now helping land managers use forest science to reduce fire risk and optimize the benefits.

CONCLUSION

From this project we came to the conclusion that decision tree has a remarkable accuracy of 99% in predicting fires in forest areas. This reduces the chances of false alarm to a great extent. Our system is able to differentiate various forest fire scenarios, from initial case (no fire) to detection of fire, fairly accurately. It can accurately determine the growth of fire. This will help in early stages of fire detection and help to confine fire to limited areas before much damage occurs. The system will be very effective in preventing occurrence of false alarms. We aim at monitoring the forests without constant human supervision.

FUTURE SCOPE

This project carries a broad prospective for future. Moreover it is a need for great research to be done in this field in the coming years. In future, our project can be extended towards finding an efficient way of localization of the fire, gravity of fire, direction of spread, area burnt and many more. In our experiment, the process of simulation of forest fire was done by burning the dried leaves directly. We could come up with ways to make this simulation more close to actual forest fires. Moreover, we can include the region.