MULTI-SCALE FOREST FIRE PREDICTION USING DEEP LEARNING

A MINOR PROJECT REPORT

Submitted by

SAYAK DAS [RA2011026010101]

ROOPAL SOOD [RA2011026010103]

Under the guidance of

Dr. M.S. ABIRAMI

(Associate Professor)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M.Nagar, Kattankulathur, Chengalpattu District

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section3 of UGC Act,1956)

BONAFIDE CERTIFICATE

Certified that 18CSP107L minor project report [18CSP108L internship report] titled **"MULTI-SCALE FOREST FIRE PREDICTION USING DEEP LEARNING"** is the bonafide work of "**SAYAK DAS [RA2011026010101], ROOPAL SOOD [RA2011026010103]**" who carried out the minor project work[internship] under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible] 16/11/23

SIGNATURE

Dr. M.S. ABIRAMI
GUIDE
Associate Professor
Dept. of Computational Intelligence

[illegible]

SIGNATURE 24/11/23

Dr. R. ANNIE UTHRA
HEAD OF THE DEPARTMENT
Professor & Head
Dept. of Computational Intelligence

[illegible] 16/11/23

Signature of the Panel Head
Dr. M.S. ABIRAMI
Associate Professor
Dept. of Computational Intelligence

ABSTRACT

Forest fires are a major threat to the environment, wildlife and to human life. They can cause widespread damage to property and infrastructure, and can also lead to loss of life. Timely and accurate prediction of forest fires is crucial for effective prevention and mitigation strategies. Traditional methods of forest fire prediction often rely on meteorological data and historical records, which may not be sufficient to capture the complex and dynamic nature of wildfires. In recent years, deep learning algorithms have emerged as a promising tool for enhancing forest fire prediction capabilities. This project aims to explore the application of deep learning techniques for forest fire prediction, leveraging the power of artificial intelligence to improve the accuracy and reliability of forecasts. The project begins by collecting and preprocessing a comprehensive dataset comprising various environmental and meteorological variables, such as temperature, humidity, wind speed, precipitation, and vegetation indices. This dataset is essential for training and evaluating the deep learning models. Additionally, historical fire incident data, including ignition locations and spread patterns, are incorporated to create a labeled dataset for supervised learning. Several deep learning algorithms, including Convolutional Neural Networks (CNNs), Long ShortTerm Memory Networks (LSTMs), and their combinations, are implemented and evaluated. CNNs are used to extract spatial patterns from satellite imagery and remote sensing data, while LSTMs are employed to model temporal dependencies in the meteorological and environmental variables. The combination of these models allows for a comprehensive analysis of both spatial and temporal factors influencing forest fire behavior. The deep learning models are evaluated using various performance metrics, including accuracy, precision, recall, and F1-score. The project also explores the use of probabilistic models to estimate the uncertainty associated with each prediction, enabling more informed decision-making by stakeholders and firefighting agencies Ultimately, the project aims to provide a reliable forest fire prediction system that can be deployed in real-time or for forecasting purposes. The system's integration into existing wildfire management infrastructure will empower authorities to make informed decisions, allocate resources effectively, and take proactive measures to mitigate the devastating impact of forest fires. The research findings from this project contribute to the growing body of knowledge in the field of artificial intelligence and environmental science and have the potential to save lives, protect ecosystems, and safeguard property from the destructive force of wildfires.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
AI	Artificial Intelligence
ML	Machine Learning
C3D	Convolutional 3 Dimensional
LSTM	Long Short-Term Memory
i.e.	id est
e.g.	exempli gratia
Eq	Equation
GPU	Graphics Processing Unit
RAM	Random-Access Memory
OpenCV	Open source Computer Vision library

CHAPTER 1

INTRODUCTION

Forest fires are a recurring natural disaster with significant ecological, environmental, and socio-economic implications. Timely and accurate prediction of forest fires plays a pivotal role in minimizing their destructive impact. In recent years, advancements in deep learning techniques have opened new avenues for enhancing prediction accuracy by harnessing the power of complex data analysis and pattern recognition.

This study focuses on the application of deep learning methodologies to predict forest fires. By integrating diverse data sources such as historical weather data, satellite imagery, and terrain characteristics, deep learning models can effectively capture spatial and temporal patterns associated with fire occurrences. The proposed deep learning model employs a convolutional neural network (CNN) to analyze satellite images and capture spatial patterns indicative of potential fire risk. The first step of the project is gathering and preprocessing a large dataset that includes a variety of meteorological and environmental variables, including temperature, humidity, wind speed, precipitation, and vegetation indices. For the deep learning models to be trained and assessed, this dataset is necessary. A labelled dataset for supervised learning is also created by incorporating historical fire incidence data, such as spread trends and ignition sites. Convolutional neural networks (CNNs), long short-term memory networks (LSTMs), and their combinations are among the deep learning methods that are put into practise and assessed. While LSTMs are utilised to model temporal relationships in the meteorological and environmental variables, CNNs are utilised to extract spatial patterns from satellite imagery and remote sensing data. The results of this study enhance the existing body of knowledge in the realms of artificial intelligence and environmental science, with the potential to mitigate the impact of wildfires, ensure ecosystem preservation, and protect lives and property from their devastating effects.

1.1 PURPOSE

The goal of this research is to transform the prediction of forest fires by utilising deep learning's capability. Deep learning offers a chance to find nuanced patterns and correlations associated to fire outbreaks because of its capacity to analyse various and complicated data sources, including satellite imagery and historical weather data.

This project aims to empower stakeholders with more accurate and useful information by creating a sophisticated prediction model. This will facilitate early warning systems, resource allocation, and proactive decision-making.

The project's primary driving force is its potential to greatly increase the accuracy of forest fire predictions, which would help to save priceless natural resources, save lives, and lessen the socioeconomic damage caused by these devastating wildfires.

1.2 OBJECTIVE

- **Multi-Scale Prediction:** Develop a predictive model that takes into account various spatial and temporal scales affecting forest fire behavior.
- **Spatial Analysis:** Utilize Convolutional Neural Networks (CNNs) to process high-resolution images and remote sensing data, enabling the extraction of spatial features and patterns contributing to fire risk.
- **Temporal Modeling:** Implement ANN and Long Short-Term Memory Networks (LSTMs) to capture temporal dependencies in meteorological variables, such as temperature, humidity, wind speed, and precipitation, at various time scales.
- **Interpretability:** Implement interpretability techniques to make the model's predictions more transparent and understandable. Highlight the key factors contributing to the predictions, enhancing usability.

1.3 FEATURE

This project introduces a groundbreaking approach to forest fire prediction by harnessing the capabilities of deep learning algorithms. The innovation lies in the integration of multiple advanced techniques to create a comprehensive and accurate prediction model.

- **Multi-modal Data Fusion**

  Deep learning methods are being used in this research study to forecast forest fires. It highlights the integration of several data kinds, including meteorological data, satellite imaging, and information from on-ground sensors, by fusing multiple modalities of data sources. The goal of this work is to improve the precision and dependability of forest fire prediction models by utilising deep learning. The ultimate objective is to create cutting-edge instruments for the early detection and control of forest fires, since they may greatly aid in the prevention of fires and the preservation of ecosystems.

- **Continuous Learning and future prediction**

  This initiative is dedicated to the ongoing enhancement of forest fire prediction through the application of deep learning methods. By harnessing sophisticated machine learning algorithms, it seeks to elevate the precision and predictive capacity of forest fire forecasting models. The ultimate objective of this project is the creation of a dependable system for forest fire prediction, providing valuable insights to advance fire prevention and ecosystem conservation.

- **Feature Engineering and Selection**

  "Feature Engineering and Selection" is the process of carefully crafting, transforming, and choosing relevant features or variables from a dataset to improve the performance of a machine learning model. It involves identifying the most informative attributes, creating new features, and eliminating redundant or less important ones to optimize model accuracy and efficiency. This crucial step is fundamental in enhancing the predictive power of machine learning models.

- **Hybrid Spatial-Temporal Modeling**

  In order to anticipate forest fires, this study uses deep learning techniques to construct a hybrid modelling approach that incorporates geographical and temporal data. The objective is to create a complete system that utilises the spatial and temporal aspects of data to boost early detection and prevention skills as well as forecasting accuracy for forest fires.

CHAPTER 2
LITERATURE SURVEY

This section consists of two major parts -Research work and Related software. Research work talks about various papers and articles that were published to make improvements regarding this field. Related software talks about the existing application that were built in relation to Forest fire prediction.

2.1. RESERCH WORK

Forest fire and smoke detection using deep learning-based learning without forgetting – Springer Open

Veerappampalayam Easwaramoorthy et. Al. [1] applied transfer learning to pretrained models, including VGG16, InceptionV3, and Xception, to maximize the utility of a smaller dataset and reduce computational complexity, all while maintaining a high level of accuracy. Notably, the Xception model achieved an impressive accuracy of 98.72%. Our models were subjected to evaluation both with and without the application of Learning without Forgetting (LwF). Without LwF, Xception attained an accuracy of 79.23% when tested on a new task, the BowFire dataset. However, with the integration of LwF, Xception demonstrated a significant performance enhancement, achieving an accuracy of 91.41% on the BowFire dataset and an outstanding 96.89% on the original dataset. This highlights the effectiveness of fine-tuning in conjunction with LwF for enhancing the performance of the original dataset.

Multi-Scale Forest Fire Recognition Model Based on Improved YOLOv5s – MDPI

Gong Chen 1 et. Al. [2] introduced several enhancements to YOLOv5 to improve its effectiveness in forest fire detection. This included the incorporation of the CA attention module to emphasize forest fire-related features, the integration of CoT to enhance the model's ability to gather fire-related data, and improvements to the loss function to promote network convergence and achieve more precise forest fire detection. Furthermore, we added the Bi-directional Feature Pyramid Network to the feature fusion layer, allowing for improved feature integration across different image scales. As a result, the model's feature fusion capabilities were enhanced. Our experimental results demonstrated that the model presented in this study achieved a mean average precision (mAP) of 87.7% and a processing speed of 36.6 frames per second (FPS). When compared to the original YOLOv5, YOLOv5s-CCAB offered a superior balance between accuracy, computational complexity (GFLOPs), and latency. These improvements significantly boosted the model's performance. Given the complexity of real forest environments and the challenge of detecting forest fires of varying scales, YOLOv5s-CCAB proves effective in timely forest fire detection and exhibits superior performance in identifying fires at different stages, particularly in their early and middle phases. In the context of forest fire detection, this model can effectively contribute to forest protection.

FOREST FIRES DETECTION AND PREDICTION USING DEEP LEARNING AND DRONES

MIMOUN YANDOUZI1 et. Al. [3] have worked on object detection within the field of computer vision, which involves the process of identifying a particular object within an image or a sequence of video frames. Object detection can be applied to locate either a single object or a group of objects . Over the recent years, numerous object detection methods, particularly those based on deep learning, have been developed . Deep learning-based object detectors have demonstrated outstanding performance on various object detection benchmarks. Notably, the YOLO algorithm, SSD algorithm, and Faster R-CNN algorithm are among the most widely recognized and utilized deep learning-based object detection techniques.

A review of machine learning applications in wildfire science and management

T Preeti et. Al. [4] developed the selection of data and models, there are two important factors to consider when specifying a model: feature selection and spatial autocorrelation. Understanding the problem domain is crucial in identifying a set of potential features. However, it's important to note that while many machine learning methods are not constrained by the number of features, having more variables doesn't necessarily translate to a more accurate, interpretable, or easily implementable model. In fact, it can lead to issues such as overfitting and increased computational time, as highlighted by Schoenberg (2016) and Breiman (2001).To address the need for selecting a reduced and more optimal set of features, two different machine learning methods come into play: Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO). Sachdeva et al. (2018) employed a GA to select input features for Boosted Regression Trees (BRT), achieving superior accuracy compared to other methods like Artificial Neural Networks (ANN), Random Forest (RF), Support Vector Machine (SVM), SVM with PSO (PSO-SVM), Decision Trees (DTs), Logistic Regression (LR), and Naive Bayes (NB).

Hong (2018) used a similar approach for fire-susceptibility mapping, resulting in improvements for both SVM and RF compared to their non-optimized versions. Tracy et al. (2018) introduced a novel random subset feature selection algorithm, which led to higher Area Under the Curve (AUC) values and reduced model complexity. Jaafari et al. (2019) combined a Neutrosophic Fuzzy Model (NFM) with the imperialist competitive algorithm (a variant of GA) for feature selection, achieving very high model accuracy (0.99) in their study. Bui et al. (2017) applied PSO to select inputs for a neural fuzzy model, resulting in improved outcomes. Zhang et al. (2019) considered the information gain ratio for feature selection.As emphasized in Moritz et al. (2012) and Mayr et al. (2018), it's important to account for spatial autocorrelation when modeling fire probabilities in a spatial context. The presence of spatial autocorrelation violates the assumption of independence in parametric models, potentially compromising the model's performance.

2.2 RELATED SOFTWARE

The "Multi-Scale Forest Fire Prediction Using Deep Learning" project employs various software and technologies, including:

- Deep Learning Frameworks: The project likely utilizes deep learning frameworks such as TensorFlow, Keras, or PyTorch for developing and training neural network models.
- Python: Python is a primary programming language for implementing machine learning and deep learning algorithms.
- Data Preprocessing Tools: Tools for data cleaning, normalization, and feature engineering, which could include Pandas, NumPy, and Scikit-Learn.
- Model Architectures: Implementation of deep learning model architectures, such as Convolutional Neural Networks (CNNs) for image analysis and recurrent neural networks (RNNs) for temporal data.
- Visualization Tools: Data visualization libraries like Matplotlib and Seaborn for creating visualizations and plots.

CHAPTER 3
SYSTEM ARCHITECTURE AND DESIGN

System architecture and design refer to the process of defining the framework, elements, modules, interfaces, and data flows of a software or hardware system. This entails crafting a plan that provides direction for the construction and execution of the system.

3.1 SYSTEM ARCHITECTURE

3.1.1 FOREST FIRE PREDICTION USING TEMPORAL DATA

A customized Artificial neural network was employed to predict forest fire from learning the temporal datas. The deep feature extraction of the input data sequence is the goal of the ANN architecture. We use the fully-connection layer's feature output as deep representation in our implementation. Next, the relationship between the input deep representations is learned using the custom ANN model. In our solution, to prevent overfitting a Dropout layer is add together to the hidden layer of the custom ANN architecture.

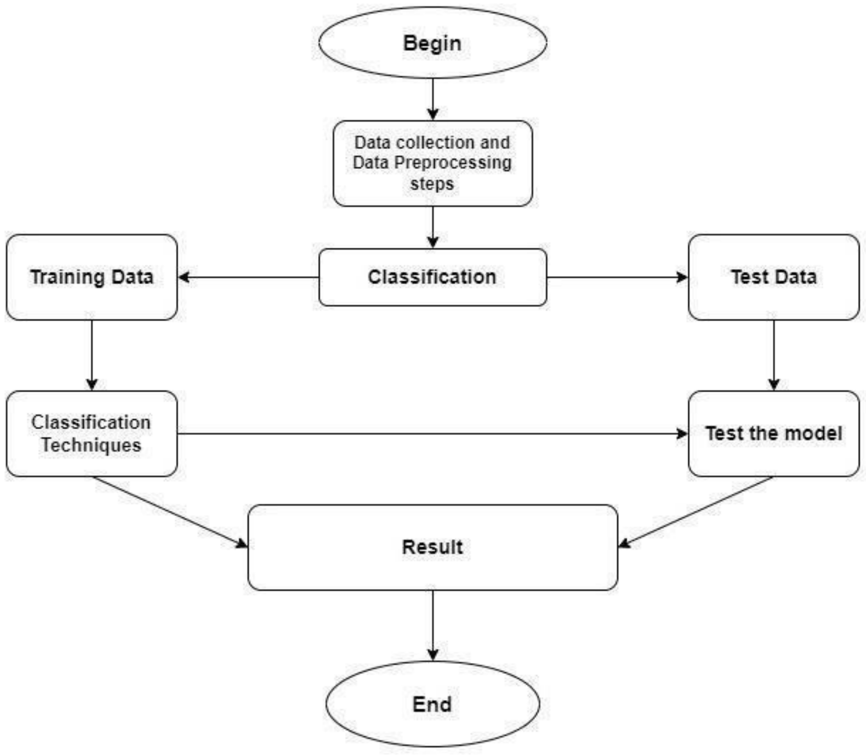

Begin
Data collection and Data Preprocessing steps
Training Data
Classification
Test Data
Classification Techniques
Test the model
Result
End

Fig 3.1: Architecture Diagram

3.1.2 FOREST FIRE PREDICTION USING IMAGE DATASET

A customized convolutional neural network(CNN) and a Recurrent-Convolutional Neural Network(RCNN) was employed to recognize the forest fire by analyzing the images. The deep feature extraction of the images is the goal of the CNN architecture. We use the fully-connection layer's feature output as deep representation in our implementation. Next, the relationship between the input deep representations is learned using the custom CNN model. In our solution, to prevent overfitting, a Dropout layer is added to the hidden layer of the custom CNN and RCNN architecture.

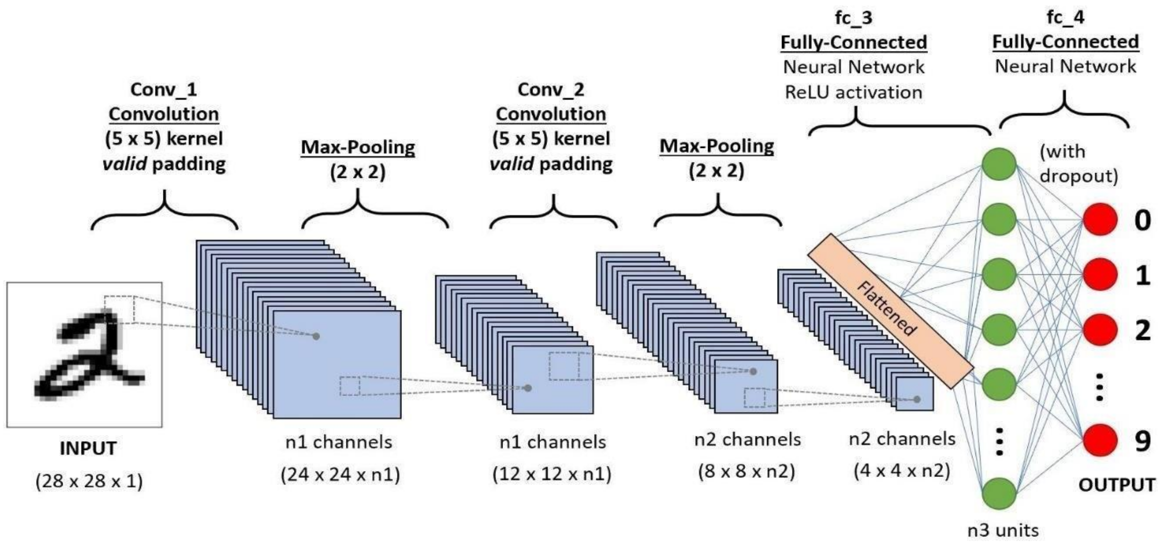


Fig 3.2 – Architecture Diagram(CNN)

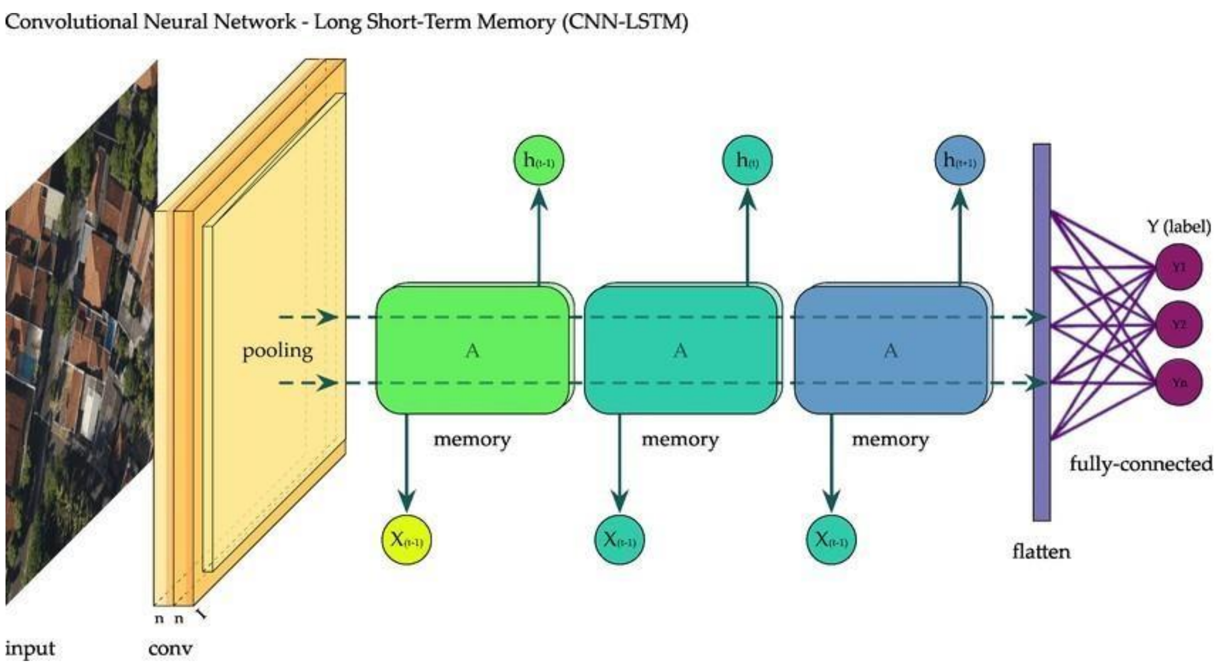
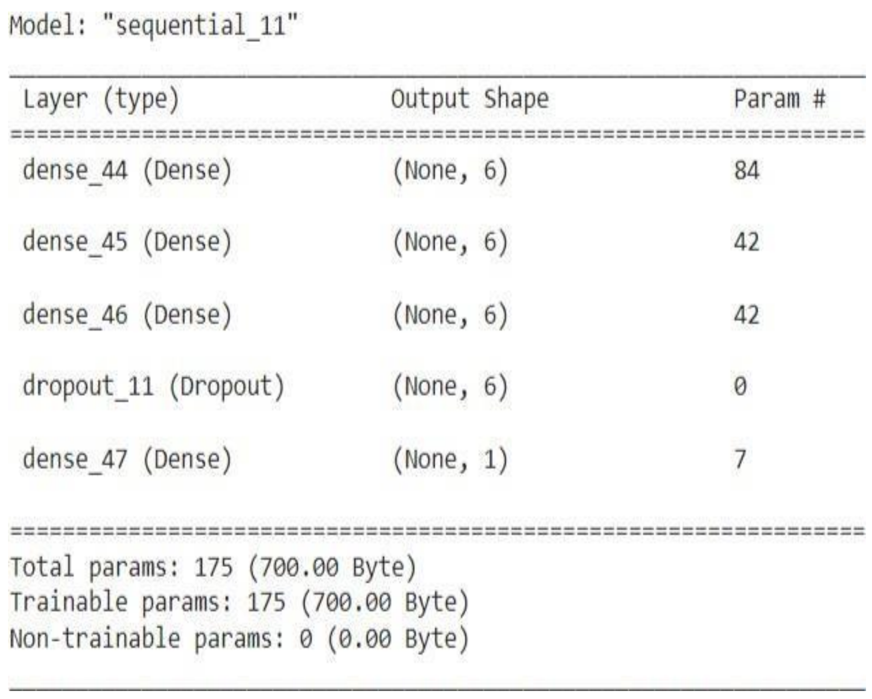


Fig 3.3 - Architecture Diagram(RCNN)

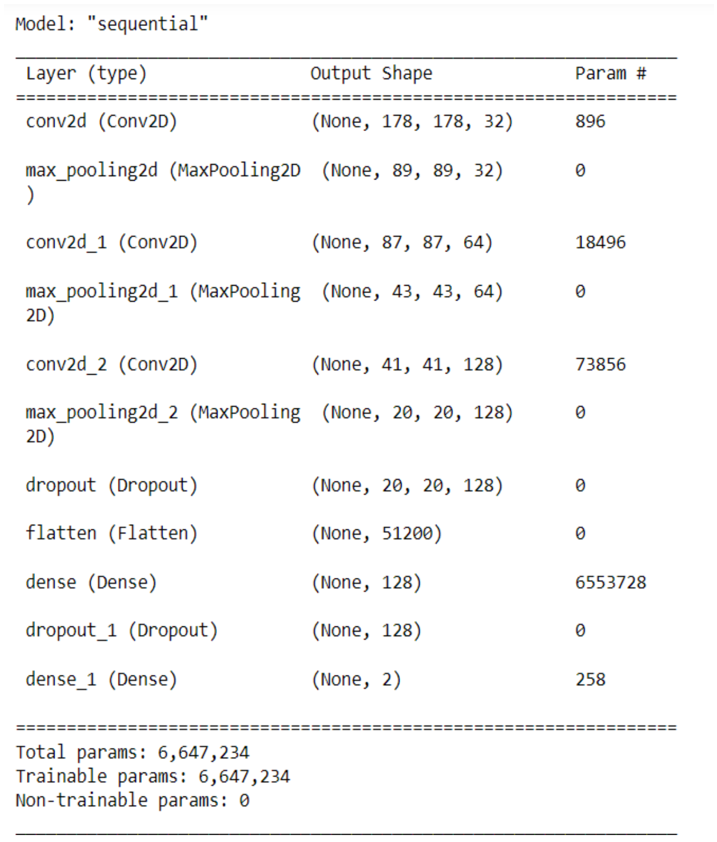
3.2 DESIGN OF MODULES

3.2.1 FOREST FIRE PREDICTION USING TEMPORAL DATA

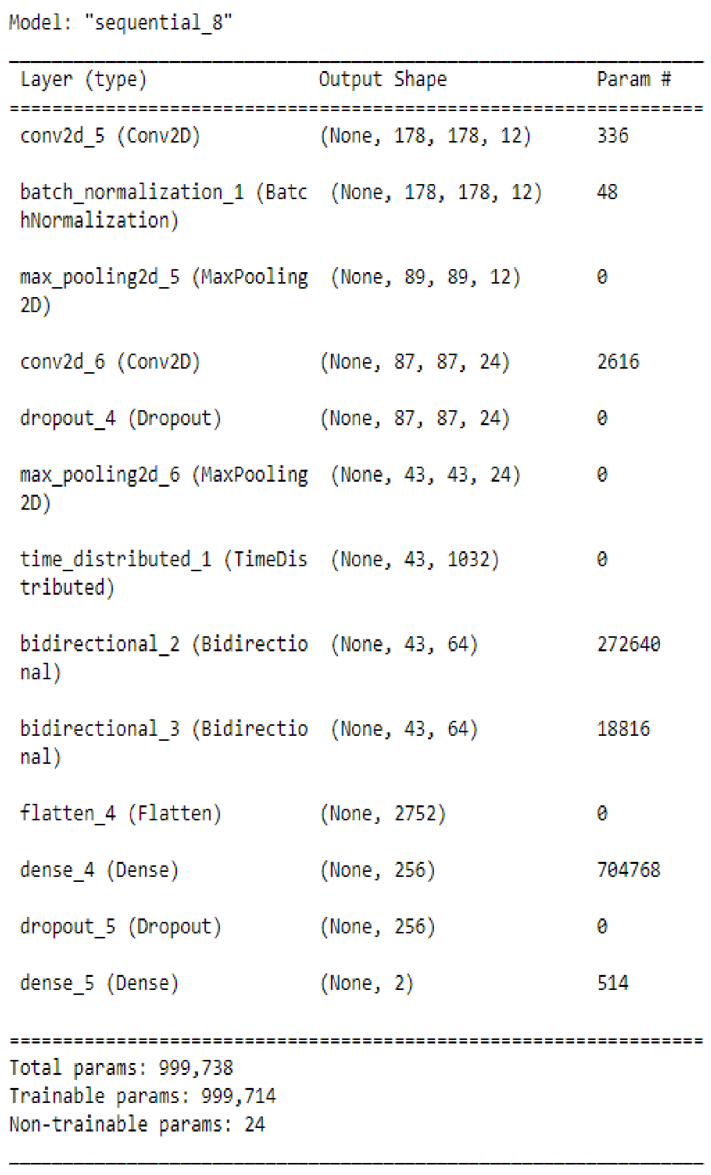


```
Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_44 (Dense)            (None, 6)                 84

 dense_45 (Dense)            (None, 6)                 42

 dense_46 (Dense)            (None, 6)                 42

 dropout_11 (Dropout)        (None, 6)                 0

 dense_47 (Dense)            (None, 1)                 7

=================================================================
Total params: 175 (700.00 Byte)
Trainable params: 175 (700.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

3.2.2 FOREST FIRE PREDICTION USING IMAGE DATASET



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 178, 178, 32)      896

 max_pooling2d (MaxPooling2D  (None, 89, 89, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 87, 87, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 43, 43, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 41, 41, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 20, 20, 128)      0
 2D)

 dropout (Dropout)           (None, 20, 20, 128)       0

 flatten (Flatten)           (None, 51200)             0

 dense (Dense)               (None, 128)               6553728

 dropout_1 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 2)                 258

=================================================================
Total params: 6,647,234
Trainable params: 6,647,234
Non-trainable params: 0
_________________________________________________________________
```



```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 178, 178, 12)      336

 batch_normalization_1 (Batc  (None, 178, 178, 12)     48
 hNormalization)

 max_pooling2d_5 (MaxPooling  (None, 89, 89, 12)       0
 2D)

 conv2d_6 (Conv2D)           (None, 87, 87, 24)        2616

 dropout_4 (Dropout)         (None, 87, 87, 24)        0

 max_pooling2d_6 (MaxPooling  (None, 43, 43, 24)       0
 2D)

 time_distributed_1 (TimeDis  (None, 43, 1032)         0
 tributed)

 bidirectional_2 (Bidirectio  (None, 43, 64)           272640
 nal)

 bidirectional_3 (Bidirectio  (None, 43, 64)           18816
 nal)

 flatten_4 (Flatten)         (None, 2752)              0

 dense_4 (Dense)             (None, 256)               704768

 dropout_5 (Dropout)         (None, 256)               0

 dense_5 (Dense)             (None, 2)                 514

=================================================================
Total params: 999,738
Trainable params: 999,714
Non-trainable params: 24
_________________________________________________________________
```

4. MULTI-SCALE FOREST FIRE PREDICTION

The forest fire prediction project employs deep learning algorithms to forecast wildfires. It begins with data collection, encompassing meteorological and environmental variables and historical fire incident records. Various deep learning models, including CNNs, RNN and LSTMs, are implemented to analyze spatial and temporal patterns in the data. Transfer learning is utilized to enhance model generalization by fine-tuning pre-trained models. Interpretability techniques and uncertainty quantification are applied for transparency. Finally, the models are rigorously evaluated, and the system is deployed for real-time or forecasting use, aiding stakeholders in proactive wildfire management and mitigation strategies.

1.1 DATASET DESCRIPTION

1.1.1 TEMPORAL DATASET

Temporal Dataset contains a CSV file which includes 516 rows of data with 13 features each. Each feature plays a vital role in the forest fire occurrence, so each data is important for the early prediction

Attribute Information:

X : x-axis spatial coordinate within the Montesinho park map: 1 to 9
Y : y-axis spatial coordinate within the Montesinho park map: 2 to 9
month : month of the year: ‘jan’ to ‘dec’
day : day of the week: ‘mon’ to ‘sun’
FFMC : FFMC (Fine Fuel Moisture Code) index from the FWI system: 18.7 to 96.20
DMC : DMC (Duff Moisture Code) index from the FWI system: 1.1 to 291.3
DC : DC (Drought Code) index from the FWI system: 7.9 to 860.6
ISI : ISI (Initial Spread Index) index from the FWI system: 0.0 to 56.10
temp : temperature in Celsius degrees: 2.2 to 33.30
RH : relative humidity in %: 15.0 to 100
wind : wind speed in km/h: 0.40 to 9.40
rain : outside rain in mm/m2 : 0.0 to 6.4
area : the burned area of the forest (in ha): 0.00 to 1090.84

1.1.2 IMAGE DATASET

Image dataset includes two folders one for Train and one for test.

Train dataset includes two folders which are labelled as Fire and No_fire Fire folder contains 2644 images of Forest fires

No_fire folder contains 2317 images of normal greenery of forest area So, Train dataset includes total 4961 images

Test dataset contains 1102 images

1.2 MODULE DESCRIPTION

1.2.1 FOREST FIRE PREDICTION USING TEMPORAL DATA

Import several Python libraries and deep learning frameworks for machine learning or deep learning.

Data import on the notebook and preprocess the data for deep learning usage
The features and labels are randomly rearranged and divided into X_train, X_test, y_train, and y_test. The test_size parameter determines the portion of data allocated to the test set, and random_state ensures that the process can be reproduced by setting a specific random seed.

The data is partitioned randomly, with 80% used for training and 20% for testing.

We create our artificial neural network (ANN) model using a Keras class called Sequential. After initializing the ANN, we proceed to establish its layers. Specifically, we construct a foundational model network comprising:

1 input layer
2 hidden layers
1 dropout layer
1 output layer

For instance, the line `model.add(Dense(6, input_dim=13, activation='relu'))` introduces the first hidden layer into the model. It consists of 6 units, expects input data with 13 features, and employs the Rectified Linear Unit (ReLU) activation function.

Multiple LSTM (Long Short-Term Memory) layers, each with 100 neurons and dropout layers (dropout rate of 0.2) for regularization. These stacked LSTM layers form a deep neural network. The model culminates with a single-unit dense output layer. The 'adam' optimizer is chosen for training, with 'binary_crossentropy' as the loss function, indicative of binary classification. This architecture aims to capture temporal dependencies in the data. Overall, it configures a deep LSTM network with dropout for mitigating overfitting in binary classification tasks.

1.2.2 FOREST FIRE PREDICTION USING IMAGE DATASET

Dataset import on the notebook and preprocess the data for deep learning usage and better visualization.

You can utilize TensorFlow along with its Keras API to generate a training dataset from an image directory. This dataset comprises images resized to 180x180 pixels and employs one-hot encoding for categorical labels. Additionally, 20% of the data is allocated for validation purposes during the training of the deep learning model.

The line `model = Sequential()` initiates a sequential model, enabling the incremental construction of a deep learning model, layer by layer.

CNN Model

1. First Convolutional Layer: This initial layer contains 32 filters with a 3x3 kernel size. It utilizes ReLU activation to introduce non-linearity and takes input images of 180x180 pixels with 3 color channels (RGB). It is followed by a max-pooling layer with a 2x2 pool size.

2. Second Convolutional Layer: Similar to the first layer but with 64 filters.

3. Third Convolutional Layer: This layer introduces another convolutional layer with 128 filters. It also incorporates a dropout layer, randomly deactivating 50% of the input units during training to prevent overfitting.

4. Flatten Layer: The purpose of this layer is to flatten the multi-dimensional output from the previous layers into a one-dimensional vector.

5. Dense Layer: Comprising 128 neurons with ReLU activation, this layer adds another level of complexity. It includes an additional dropout layer with a 25% dropout rate.

6. Dense Layer with Softmax Activation: Serving as the output layer, this layer has a number of neurons equal to the number of classes, as determined by `len(class_names)`. It employs the softmax activation function, which scales the model's outputs into probabilities, making it well-suited for classification tasks.

RCNN Model

1. First Convolutional Layer:

- This initial layer features 12 filters with a 3x3 kernel size, applying ReLU activation.
- It incorporates batch normalization to standardize the activations from the previous layer.
- Subsequently, a max-pooling operation with a 2x2 pool size is employed.

2. Second Convolutional Layer:

- The second layer consists of 24 filters with a 3x3 kernel, utilizing ReLU activation.
- It introduces a dropout layer with a rate of 0.2, allowing for the random deactivation of 20% of the units.
- Following the dropout layer, there's another max-pooling operation with a 2x2 pool size.

TimeDistributed Layer with Bidirectional LSTM and GRU Layers:

The TimeDistributed layer is used to apply the same operation to each time step of the input sequence.

Bidirectional LSTM and GRU layers with 32 units are used to capture temporal dependencies in the data. These layers return sequences and have dropout and recurrent dropout for regularization.

Flatten Layer: This layer flattens the output from the time-distributed layers.

Dense Layers:

A dense layer of 256 unit neurons and ReLU activation is attached .A dropout layer with a rate of 0.5 is added.

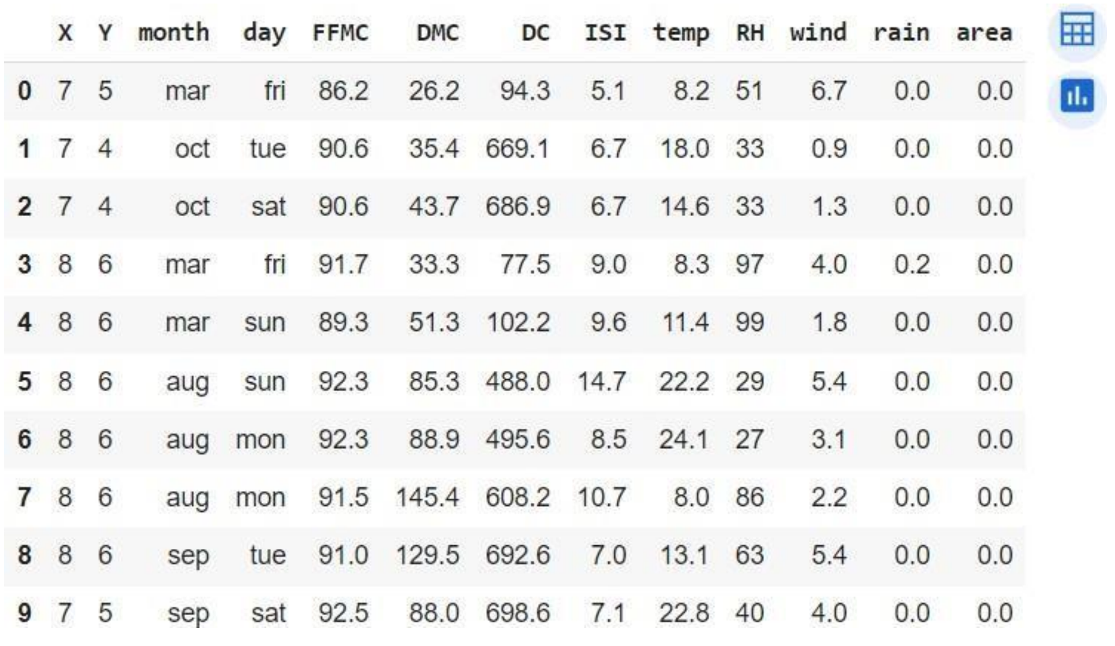
The final dense layer has neurons equal to the number of classes (represented by len(class_names)) and uses softmax activation for classification.

5. CODING AND TESTING

5.1 FOREST FIRE PREDICTION USING TEMPORAL DATA

```
#import required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('seaborn')
import seaborn as sns
from sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
import tensorflow as tensorflow
from keras.models import Sequential
from keras.layers import Dense, Dropout
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.utils import to_categorical
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
```

```
#import dataset from google drive
df = pd.read_csv("/content/gdrive/My Drive/Datasets/Forest Fire dataset/forestfires.csv")
df.head(10)  #Gives first ten rows of the dataset
```



	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0
5	8	6	aug	sun	92.3	85.3	488.0	14.7	22.2	29	5.4	0.0	0.0
6	8	6	aug	mon	92.3	88.9	495.6	8.5	24.1	27	3.1	0.0	0.0
7	8	6	aug	mon	91.5	145.4	608.2	10.7	8.0	86	2.2	0.0	0.0
8	8	6	sep	tue	91.0	129.5	692.6	7.0	13.1	63	5.4	0.0	0.0
9	7	5	sep	sat	92.5	88.0	698.6	7.1	22.8	40	4.0	0.0	0.0

Fig 5.1 - Glimpse of dataset

```
df['size_category'] = np.where(df['area']>6, '1', '0')
df['size_category']= pd.to_numeric(df['size_category'])
df.tail(10)

df['size_category'].value_counts()

NO_Fire    378
Fire    139
Name: size_category, dtype: int64

sns.countplot(x='size_category', data=df, palette="tab10")
plt.title('Class Distributions \n 0: No Fire || 1: Fire', fontsize=14)
#plotting PieChart
classlabels = ["FIRE", "NOT FIRE"]
plt.figure(figsize =(12, 7))
plt.pie(percentage,labels = classlabels,autopct='%1.1f%%')
plt.title ("Pie Chart of Classes", fontsize = 15)
plt.show()
```

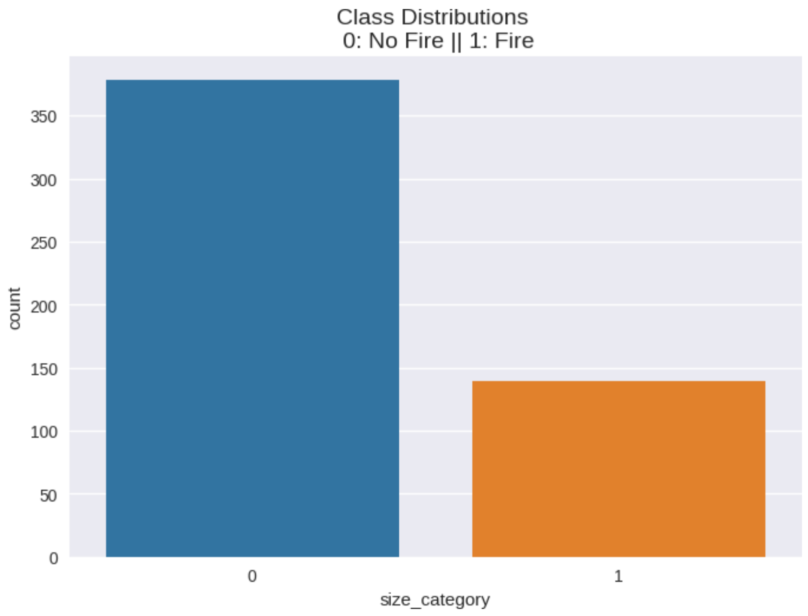


Fig 5.2- fire and non-fire graph

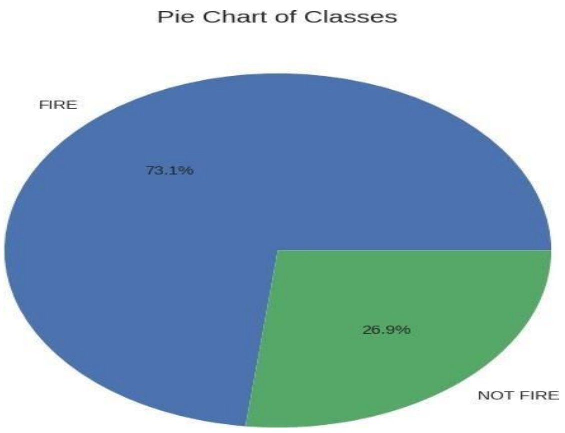
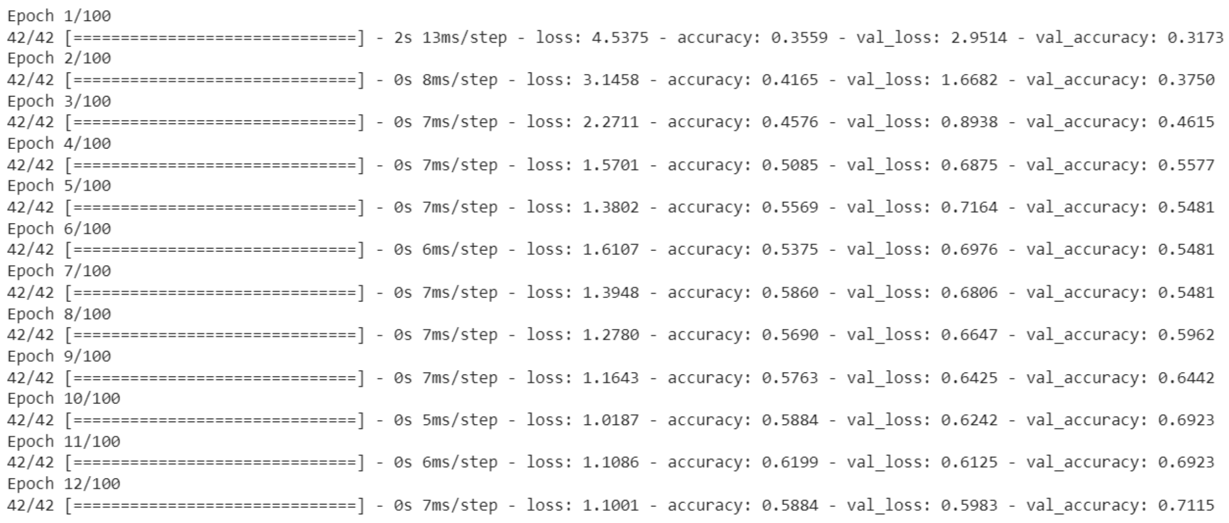


Fig 5.3 – pie chart of dataset

```
# fitting scaler
sc_features = StandardScaler()
# transforming features
X_test = sc_features.fit_transform(X_test)
X_train = sc_features.transform(X_train)
# features
X_test = pd.DataFrame(X_test, columns = features.columns)
X_train = pd.DataFrame(X_train, columns = features.columns)
# labels
y_test = pd.DataFrame(y_test, columns = ['size_category'])
y_train = pd.DataFrame(y_train, columns = ['size_category'])
X_train.head()

model = Sequential()
model.add(Dense(6, input_dim=13, activation='relu'))
model.add(Dense(6, activation='relu'))
model.add(Dense(6, activation='sigmoid'))
model.add(Dropout(0.2))
model.add(Dense(1, activation = 'relu'))
model.summary()
# Compile Model
model.compile(optimizer = 'adam', metrics=['accuracy'], loss ='binary_crossentropy')
# Train Model
history = model.fit(X_train, y_train, validation_data = (X_test, y_test), batch_size = 10,
epochs = 100)
```



```
Epoch 1/100
42/42 [==============================] - 2s 13ms/step - loss: 4.5375 - accuracy: 0.3559 - val_loss: 2.9514 - val_accuracy: 0.3173
Epoch 2/100
42/42 [==============================] - 0s 8ms/step - loss: 3.1458 - accuracy: 0.4165 - val_loss: 1.6682 - val_accuracy: 0.3750
Epoch 3/100
42/42 [==============================] - 0s 7ms/step - loss: 2.2711 - accuracy: 0.4576 - val_loss: 0.8938 - val_accuracy: 0.4615
Epoch 4/100
42/42 [==============================] - 0s 7ms/step - loss: 1.5701 - accuracy: 0.5085 - val_loss: 0.6875 - val_accuracy: 0.5577
Epoch 5/100
42/42 [==============================] - 0s 7ms/step - loss: 1.3802 - accuracy: 0.5569 - val_loss: 0.7164 - val_accuracy: 0.5481
Epoch 6/100
42/42 [==============================] - 0s 6ms/step - loss: 1.6107 - accuracy: 0.5375 - val_loss: 0.6976 - val_accuracy: 0.5481
Epoch 7/100
42/42 [==============================] - 0s 7ms/step - loss: 1.3948 - accuracy: 0.5860 - val_loss: 0.6806 - val_accuracy: 0.5481
Epoch 8/100
42/42 [==============================] - 0s 7ms/step - loss: 1.2780 - accuracy: 0.5690 - val_loss: 0.6647 - val_accuracy: 0.5962
Epoch 9/100
42/42 [==============================] - 0s 7ms/step - loss: 1.1643 - accuracy: 0.5763 - val_loss: 0.6425 - val_accuracy: 0.6442
Epoch 10/100
42/42 [==============================] - 0s 5ms/step - loss: 1.0187 - accuracy: 0.5884 - val_loss: 0.6242 - val_accuracy: 0.6923
Epoch 11/100
42/42 [==============================] - 0s 6ms/step - loss: 1.1086 - accuracy: 0.6199 - val_loss: 0.6125 - val_accuracy: 0.6923
Epoch 12/100
42/42 [==============================] - 0s 7ms/step - loss: 1.1001 - accuracy: 0.5884 - val_loss: 0.5983 - val_accuracy: 0.7115
```

```
plt.figure(figsize=[8,5])
plt.plot(history.history['accuracy'], label='Train')
plt.plot(history.history['val_accuracy'], label='Valid')
plt.legend()
plt.xlabel('Epochs', fontsize=16)
plt.ylabel('Accuracy', fontsize=16)
plt.title('Accuracy Curves Epoch 100, Batch Size 10', fontsize=16)
plt.show()
```

]

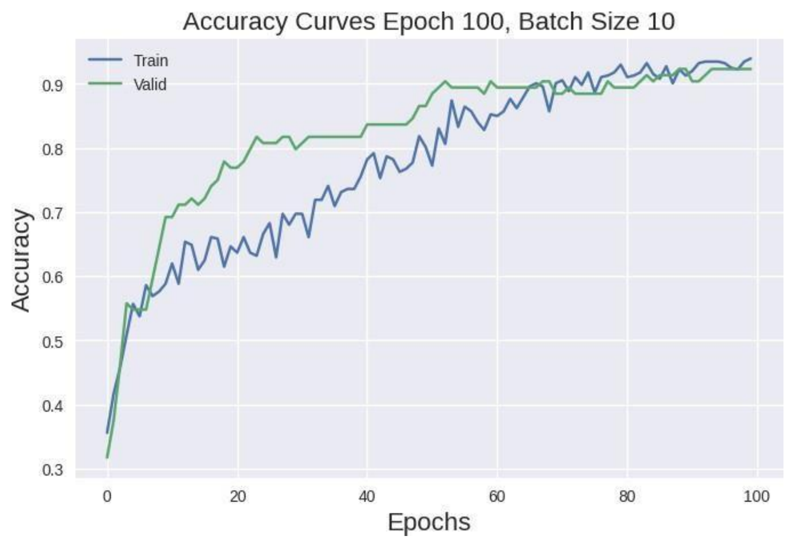


Fig 5.4 – Accuracy graph

```
#Batch Size = 6, Early Stopping (Patience, Model Checkpoint)
# Define a function to initialize the model
def init_model():
    # Define model
    model = Sequential()
    model.add(Dense(6, input_dim=13, activation='relu'))
    model.add(Dense(6, activation='relu'))
    model.add(Dense(6, activation='sigmoid'))
    model.add(Dropout(0.2))
    model.add(Dense(1, activation='relu'))

    # Compile model
    model.compile(optimizer='adam',
                  metrics=['accuracy'],
                  loss='binary_crossentropy')
    return model

# Initialize the model
model = init_model()

# Define early stopping and model checkpoint callbacks
es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=150)
mc = ModelCheckpoint('best_model.h5', monitor='val_accuracy', mode='max', verbose=1,
save_best_only=True)

# Fit the model with callbacks
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=250,
verbose=0, batch_size=6, callbacks=[es, mc])
```

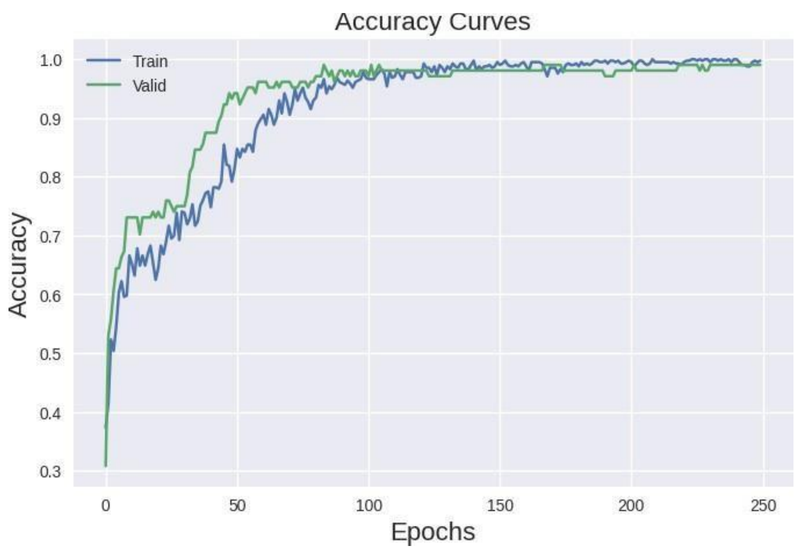


Fig 5.5 – Accuracy graph (Earlystop)

```
model_2 = Sequential()

model_2.add(LSTM(units=100, return_sequences = True, input_shape
=(X_train.shape[1],1)))
model_2.add(Dropout(0.2))

model_2.add(LSTM(units=100, return_sequences = True))
model_2.add(Dropout(0.2))

model_2.add(LSTM(units=100, return_sequences = True))
model_2.add(Dropout(0.2))

model_2.add(LSTM(units=100, return_sequences = False))
model_2.add(Dropout(0.2))

model_2.add(Dense(units =1))

model_2.compile(optimizer = 'adam', metrics=['accuracy'], loss ='binary_crossentropy')
model_2.summary()
```

5.2 FOREST FIRE PREDICTION USING IMAGE DATASET

```
#GENERAL
import pandas as pd
import numpy as np
import pathlib
import PIL
import os
import seaborn as sns
import matplotlib.pyplot as plt
#PATH PROCESS
import os
import os.path
from pathlib import Path
import glob
#IMAGE PROCESS
from PIL import Image
import tensorflow as tf
from tensorflow import keras
from keras.preprocessing import image
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.callbacks import ModelCheckpoint,EarlyStopping
#SCALER & TRANSFORMATION
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from keras.utils import np_utils
from sklearn.model_selection import train_test_split
from keras import regularizers
from sklearn.preprocessing import LabelEncoder
#ACCURACY CONTROL
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report,
roc_auc_score, roc_curve
from sklearn.model_selection import GridSearchCV, cross_val_score
from sklearn.metrics import mean_squared_error, r2_score
#OPTIMIZER
from keras.optimizers import RMSprop,Adam,Optimizer,Optimizer
#MODEL LAYERS
from tensorflow.keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D,
BatchNormalization,MaxPooling2D,BatchNormalization,\
                    Permute, TimeDistributed, Bidirectional,GRU, SimpleRNN, LSTM,
GlobalAveragePooling2D, SeparableConv2D
from keras import models

data_train = pathlib.Path("C:\\Users\\dassa\\Desktop\\Minor Project\\image
dataset\\forest_dataset\\data\\train")
data_test = pathlib.Path("C:\\Users\\dassa\\Desktop\\Minor Project\\image
dataset\\forest_dataset\\data\\test")
```

```
image_extensions = ['.jpg', '.png', '.jpeg', '.gif']

#Train Image count
image_count_train = len(list(data_train.glob('*/*.jpg')))
print(f"Number of images in the train dataset: {image_count_train}")

#Test Image count
data_count_test = [f for f in data_test.glob('*') if f.suffix in image_extensions]

# Get the count of image files
image_count = len(data_count_test)

print(f"Number of images in the test dataset: {image_count}")
```

```
-> Number of images in the train dataset: 4961
   Number of images in the test dataset: 1102
```

```
image_dataset =
tf.keras.preprocessing.image_dataset_from_directory(data_train,batch_size=32,image_size=(1
80,180),
                                              label_mode='categorical',seed=123)

class_names = image_dataset.class_names

#Dictionary to store the path of image as per the class
files_path_dict = {}

for c in  class_names:
   files_path_dict[c] = list(map(lambda
x:str(data_train)+'/'+c+'/'+x,os.listdir(str(data_train)+'/'+c)))

#Visualize image
plt.figure(figsize=(15,15))
index = 0
for c in class_names:
   path_list = files_path_dict[c][:1]
   index += 1
   plt.subplot(3,3,index)
   plt.imshow(load_img(path_list[0],target_size=(180,180)))
   plt.title(c)
   plt.axis("off")
```

```
Found 4961 files belonging to 2 classes.
```

Fig 5.6 - dataset Images

```
#Visualize the Number of image in each class.
import seaborn as sns
plt.figure(figsize=(10, 8))
sns.barplot(x="No. of Image", y="Class", data=df,
            label="Class")
```

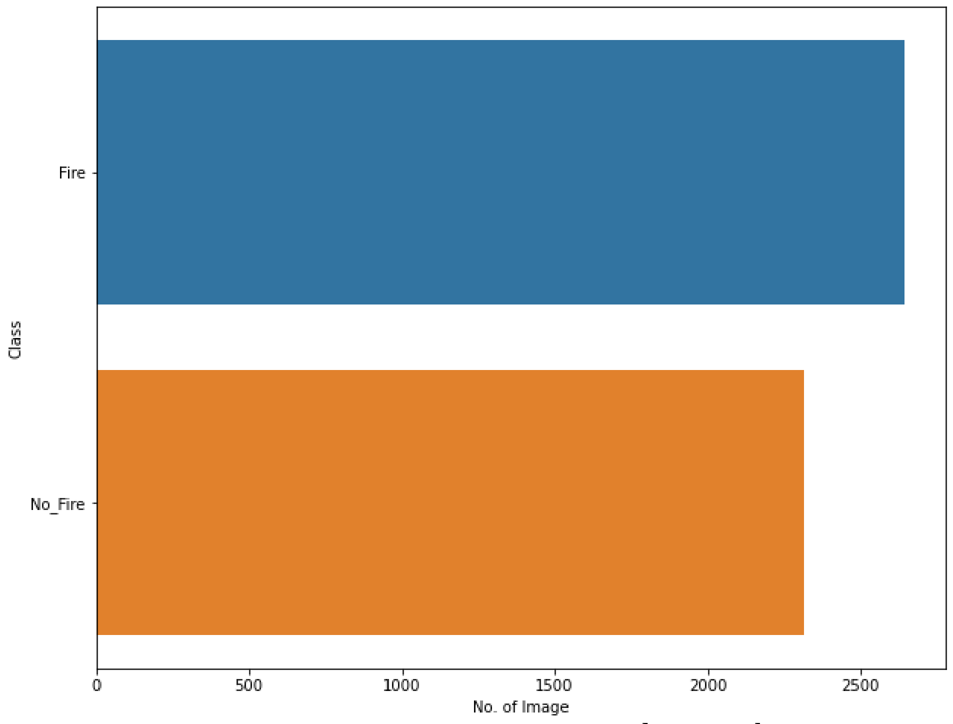
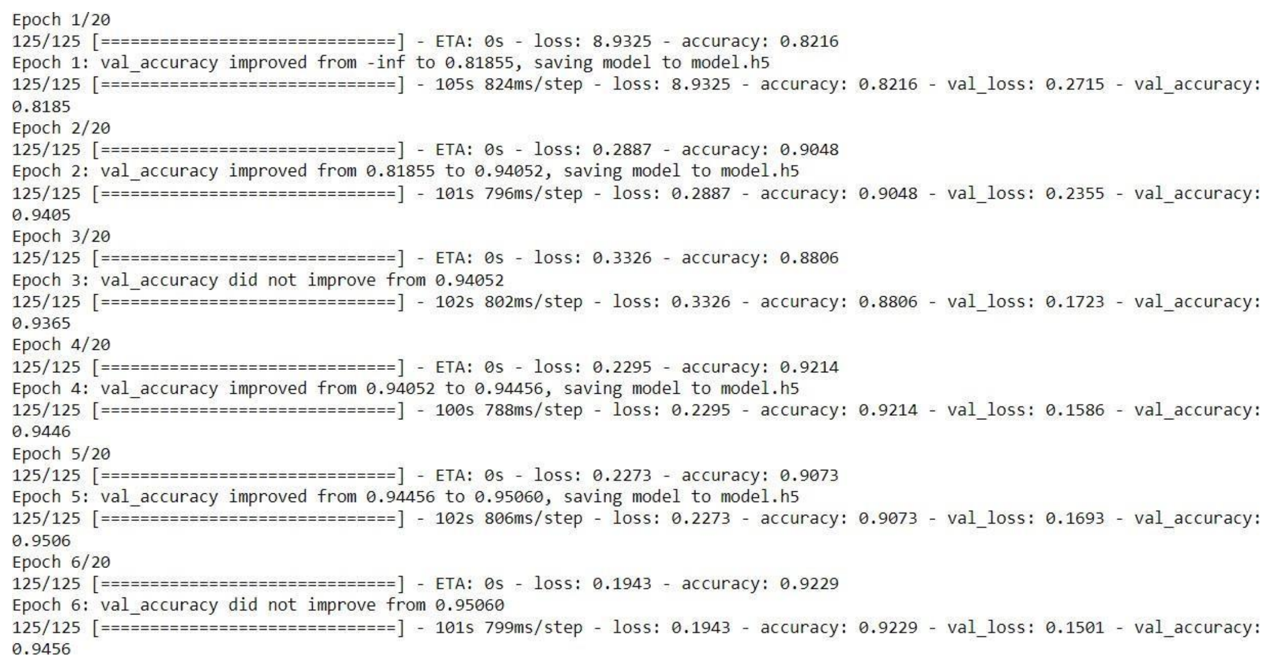


Fig 5.7- class graph

```
#CNN Model Architecture

#Sequential allows you to create models layer-by-layer
model = Sequential()


#First Convulation layer
model.add(layers.Conv2D(32,kernel_size=(3,3),activation='relu',input_shape=(180, 180, 3)))
model.add(layers.MaxPool2D(pool_size=(2,2)))

#Second Convulation Layer
model.add(layers.Conv2D(64,kernel_size=(3,3),activation='relu'))
model.add(layers.MaxPool2D(pool_size=(2,2)))
```

```
#Third Convulation Layer
model.add(layers.Conv2D(128,kernel_size=(3,3),activation='relu'))
model.add(layers.MaxPool2D(pool_size=(2,2)))

#Dropout layer with 50% Fraction of the input units to drop.
model.add(layers.Dropout(0.5))

#Flatten Layer
##Keras.layers.flatten function flattens the multi-dimensional input tensors into a single
dimension.
model.add(layers.Flatten())

#Dense Layer
model.add(layers.Dense(128,activation='relu'))

#Dropout layer with 25% Fraction of the input units to drop.
model.add(layers.Dropout(0.25))

#Dense Layer with softmax activation function.
#Softmax is an activation function that scales numbers/logics into probabilities.
model.add(layers.Dense(len(class_names),activation='softmax'))

model.compile(optimizer="Adam",loss="categorical_crossentropy",metrics=["accuracy"])
checkpoint =
ModelCheckpoint("model.h5",monitor="val_accuracy",save_best_only=True,mode="auto",ve
rbose=1)
#Stop training when a monitored metric has stopped improving.
earlystop = EarlyStopping(monitor="val_accuracy",patience=5,mode="auto",verbose=1)

# Train the model
epochs = 20
train_model = model.fit(train_ds, validation_data=val_ds,
epochs=epochs,callbacks=[checkpoint,earlystop])
```



```
Epoch 1/20
125/125 [==============================] - ETA: 0s - loss: 8.9325 - accuracy: 0.8216
Epoch 1: val_accuracy improved from -inf to 0.81855, saving model to model.h5
125/125 [==============================] - 105s 824ms/step - loss: 8.9325 - accuracy: 0.8216 - val_loss: 0.2715 - val_accuracy:
0.8185
Epoch 2/20
125/125 [==============================] - ETA: 0s - loss: 0.2887 - accuracy: 0.9048
Epoch 2: val_accuracy improved from 0.81855 to 0.94052, saving model to model.h5
125/125 [==============================] - 101s 796ms/step - loss: 0.2887 - accuracy: 0.9048 - val_loss: 0.2355 - val_accuracy:
0.9405
Epoch 3/20
125/125 [==============================] - ETA: 0s - loss: 0.3326 - accuracy: 0.8806
Epoch 3: val_accuracy did not improve from 0.94052
125/125 [==============================] - 102s 802ms/step - loss: 0.3326 - accuracy: 0.8806 - val_loss: 0.1723 - val_accuracy:
0.9365
Epoch 4/20
125/125 [==============================] - ETA: 0s - loss: 0.2295 - accuracy: 0.9214
Epoch 4: val_accuracy improved from 0.94052 to 0.94456, saving model to model.h5
125/125 [==============================] - 100s 788ms/step - loss: 0.2295 - accuracy: 0.9214 - val_loss: 0.1586 - val_accuracy:
0.9446
Epoch 5/20
125/125 [==============================] - ETA: 0s - loss: 0.2273 - accuracy: 0.9073
Epoch 5: val_accuracy improved from 0.94456 to 0.95060, saving model to model.h5
125/125 [==============================] - 102s 806ms/step - loss: 0.2273 - accuracy: 0.9073 - val_loss: 0.1693 - val_accuracy:
0.9506
Epoch 6/20
125/125 [==============================] - ETA: 0s - loss: 0.1943 - accuracy: 0.9229
Epoch 6: val_accuracy did not improve from 0.95060
125/125 [==============================] - 101s 799ms/step - loss: 0.1943 - accuracy: 0.9229 - val_loss: 0.1501 - val_accuracy:
0.9456
```

```
# Plot the training curves

epochs_range = range(earlystop.stopped_epoch+1)

plt.figure(figsize=(15, 10))
plt.subplot(1, 2, 1)

#Plot Model Accuracy
plt.plot(train_model.history['accuracy'])
plt.plot(train_model.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel(epochs_range)
plt.legend(['train', 'val'], loc='upper left')

#Plot Model Loss
plt.subplot(1, 2, 2)
plt.plot(train_model.history['loss'])
plt.plot(train_model.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel(epochs_range)
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

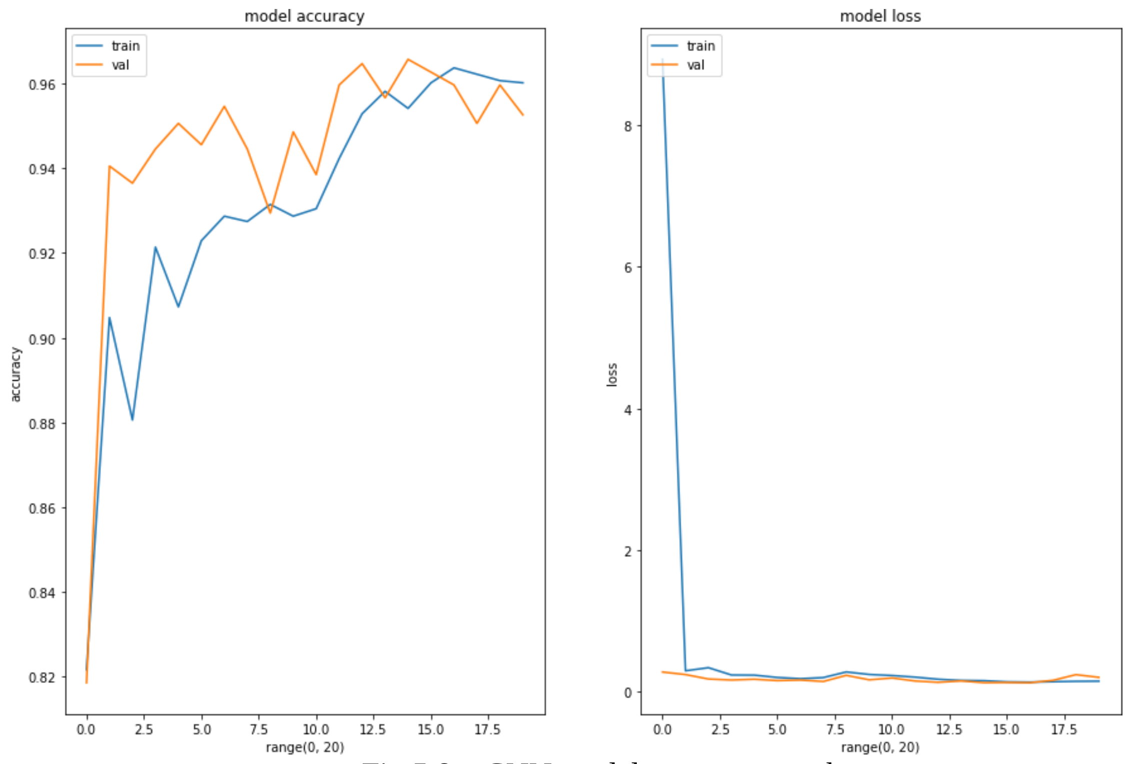


Fig 5.8 – CNN model accuracy graphs

```
_, train_acc = model.evaluate(train_ds, verbose=1)
_, valid_acc = model.evaluate(val_ds, verbose=1)
print('Train: %.3f, Valid: %.3f' % (train_acc, valid_acc))

from glob import glob
Test_image_path = os.path.join(data_test, class_names[1], '*')
Test_image = glob("C:\\Users\\dassa\\Downloads\\download.jpeg")
Test_image = load_img(Test_image[-1],target_size=(180,180,3))
plt.imshow(Test_image)
plt.grid(False)
```

```
   125/125 [==============================] - 27s 209ms/step - loss: 0.1118
- accuracy: 0.9635
   31/31 [==============================] - 7s 194ms/step - loss: 0.1961 -
accuracy: 0.9526
   Train: 0.963, Valid: 0.953
```

```
img = np.expand_dims(Test_image,axis=0)
pred = model.predict(img)
pred = np.argmax(pred)
pred_class = class_names[pred]
print( "Predictive Class "+pred_class )
```

```
#CNN-RCNN model Architecture

Model_Three = Sequential()

Model_Three.add(Conv2D(12,(3,3),activation="relu",input_shape=(180,180,3)))
Model_Three.add(BatchNormalization())
Model_Three.add(MaxPooling2D((2,2)))

#
Model_Three.add(Conv2D(24,(3,3),
        activation="relu"))
Model_Three.add(Dropout(0.2))
Model_Three.add(MaxPooling2D((2,2)))


#
Model_Three.add(TimeDistributed(Flatten()))
Model_Three.add(Bidirectional(LSTM(32,
                return_sequences=True,
                dropout=0.5,
                recurrent_dropout=0.5)))
Model_Three.add(Bidirectional(GRU(32,
                return_sequences=True,
                dropout=0.5,
                recurrent_dropout=0.5)))

#
Model_Three.add(Flatten())

Model_Three.add(Dense(256,activation="relu"))
Model_Three.add(Dropout(0.5))
Model_Three.add(Dense(len(class_names),activation="softmax"))

Model_Three.compile(optimizer = 'adam' , loss = 'categorical_crossentropy' , metrics =
['accuracy'])
checkpoint =
ModelCheckpoint("model.h5",monitor="val_accuracy",save_best_only=True,mode="auto",ve
rbose=1)

#Stop training when a monitored metric has stopped improving.
earlystop = EarlyStopping(monitor="val_accuracy",patience=5,mode="auto",verbose=1)
RCNN_Model =
Model_Three.fit(train_ds,validation_data=val_ds,callbacks=[checkpoint,earlystop],epochs=20
)

# Plot the training curves

epochs_range = range(earlystop.stopped_epoch+1)
```

```
plt.figure(figsize=(15, 10))
plt.subplot(1, 2, 1)

#Plot Model Accuracy
plt.plot(RCNN_Model.history['accuracy'])
plt.plot(RCNN_Model.history['val_accuracy'])
plt.title('model  accuracy')
plt.ylabel('accuracy')
plt.xlabel(epochs_range)
plt.legend(['train', 'val'], loc='upper left')

#Plot Model Loss
plt.subplot(1, 2, 2)
plt.plot(RCNN_Model.history['loss'])
plt.plot(RCNN_Model.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel(epochs_range)
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

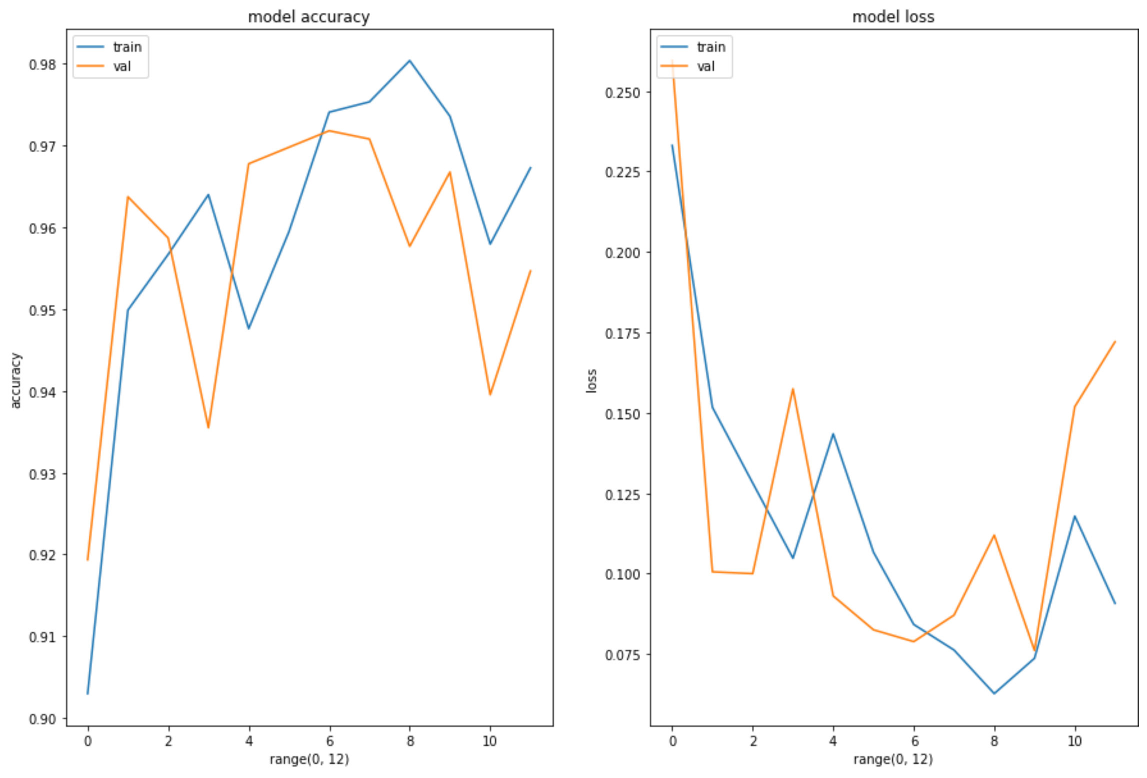


Fig 5.9 – RCNN model accuracy graph

```
from glob import glob
Test_image_path = os.path.join(data_test, class_names[1], '*')
Test_image = glob("C:\\Users\\dassa\\Downloads\\download.jpeg")
Test_image = load_img(Test_image[-1],target_size=(180,180,3))
plt.imshow(Test_image)
plt.grid(False)

img = np.expand_dims(Test_image,axis=0)
pred = model.predict(img)
pred = np.argmax(pred)
pred_class = class_names[pred]
print( "Predictive Class "+pred_class )
```

6. RESULT & DISCUSSION

6.1 FOREST FIRE PREDICTION USING TEMPORAL DATA

This research includes a comparison between three model base ANN, ANN (earlystopping) and LSTM for finding the highest accuracy. The results are as follows-

S.no	**Model**	**Train_Acc**	**Val_Acc**
1	Base ANN	96.9	92.3
2	ANN (earlystopping)	99.99	99.0
3	LSTM	73.4	73.1

Table 6.1.1 - ANN,ANN(es), LSTM accuracy comparison

From this above table we came to a conclusion that ANN with earlystopping can validate and predict accurately than other Deep learning models.

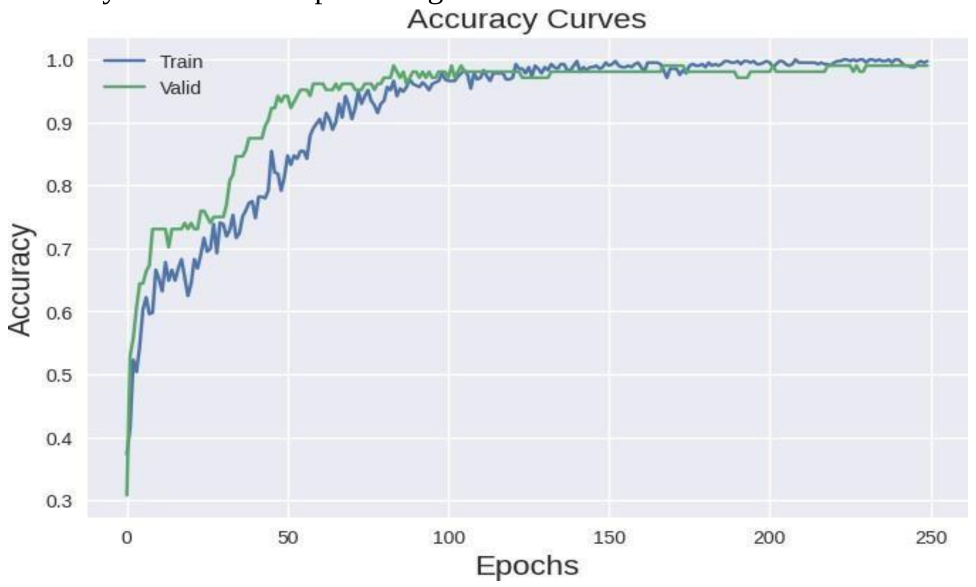


Fig – 6.1.1 – ANN model (earlystop) graph

6.2 FOREST FIRE PREDICTION USING IMAGE DATASET

This research includes a comparison between two models CNN and RCNN for finding the highest accuracy. The results are as follows-

S.no	**Model**	**Train_Acc**	**Val_Acc**
1	CNN	96.3	95.3
2	RCNN(CNN+LSTM+GRU)	97.17	96.7

Table 6.1.2 - CNN and RCNN accuracy comparison

By comparing above table we can conclude that RCNN model works efficiently and predict forest fire more precisely than CNN model.

CNN

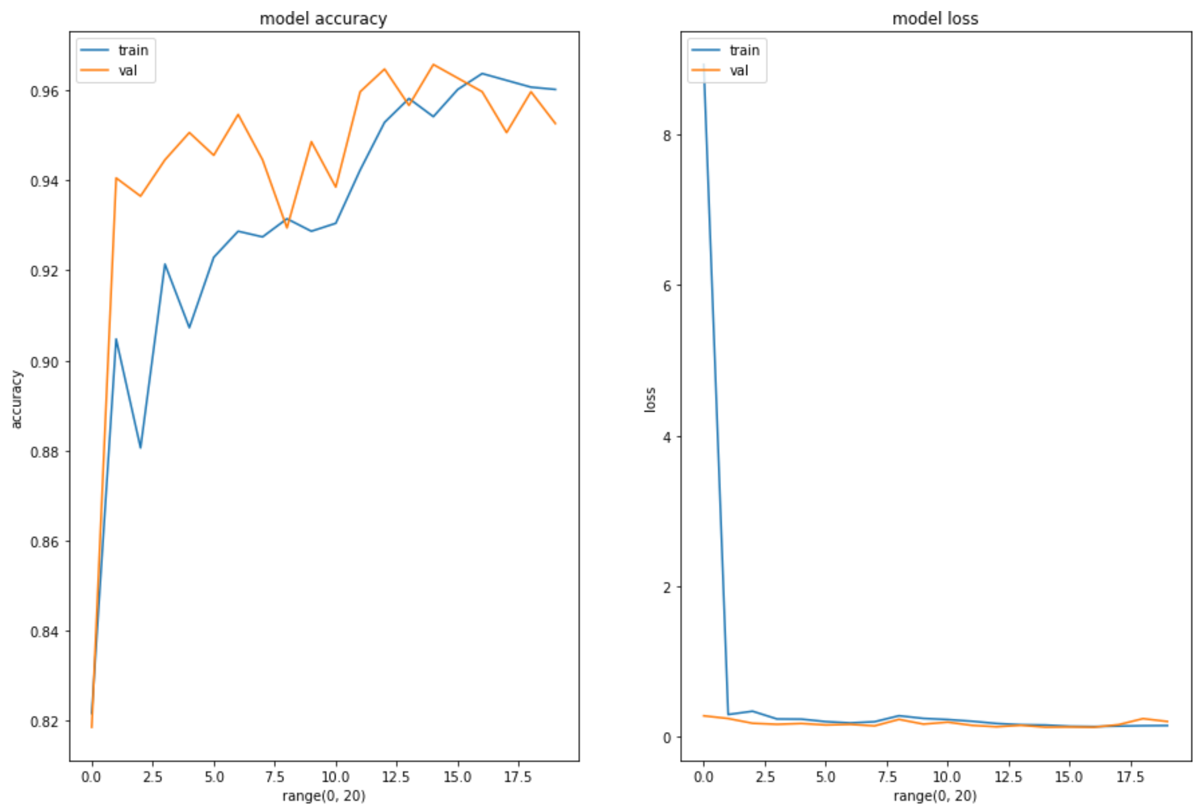


Fig 6.1.2 – CNN accuracy and loss graph

RCNN

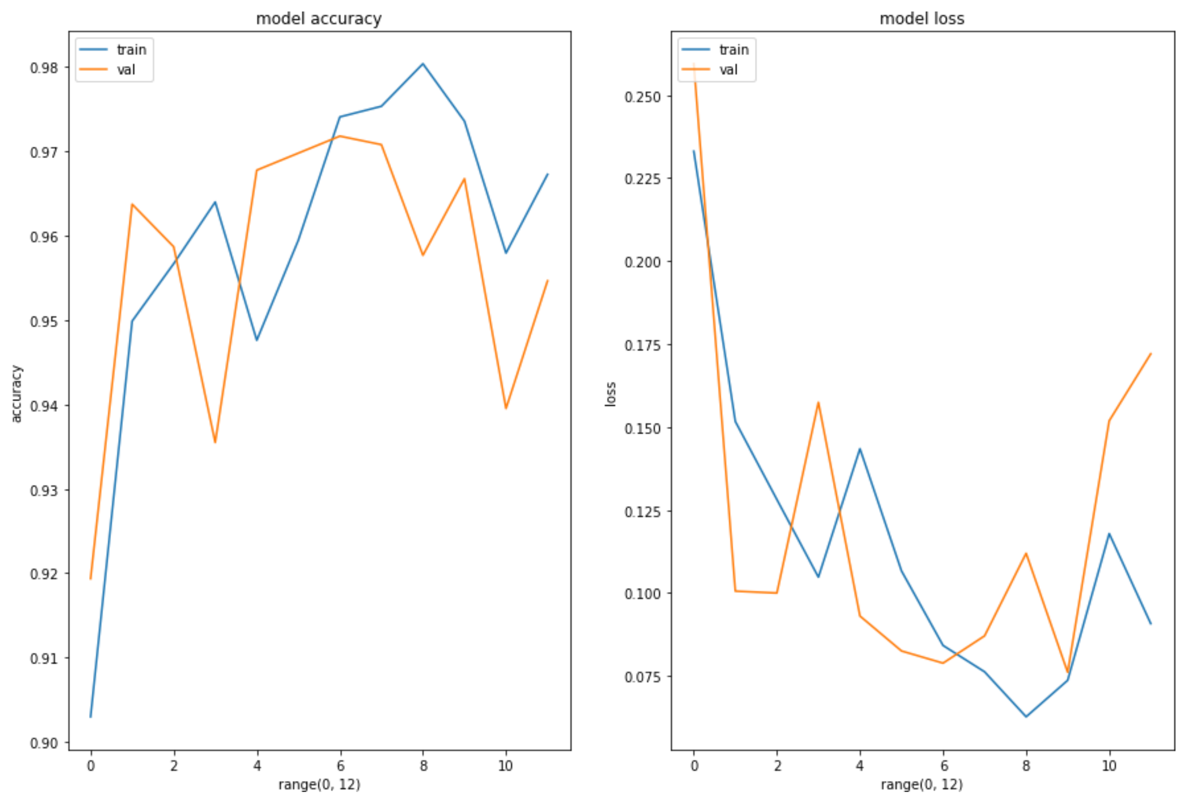
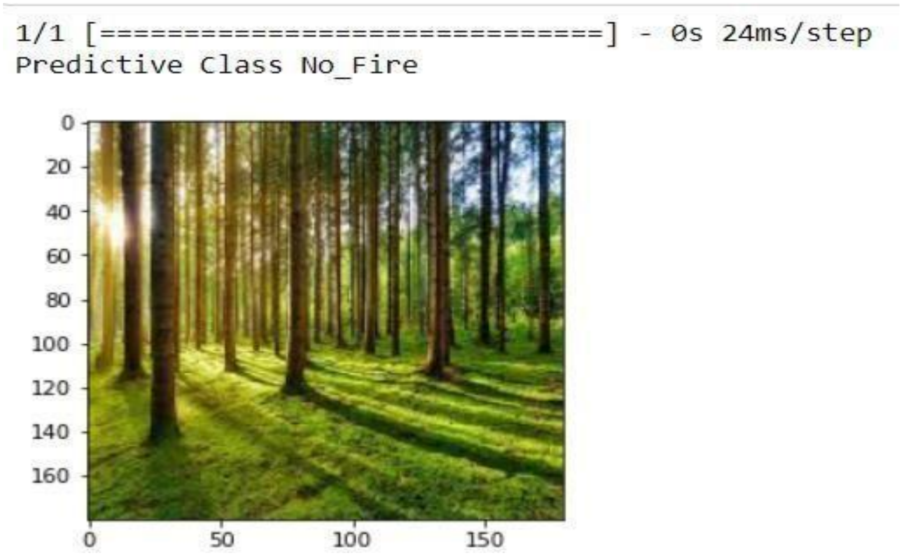


Fig 6.1.3 – RCNN accuracy and loss graph

Fig 6.1.2 – we can observe the graph and make a conclusion that CNN Model’s train accuracy is 96% and validation accuracy is 95.4%

Fig 6.1.3 – we can observe the graph and make a conclusion that RCNN Model’s train accuracy is 97.1% and validation accuracy is 96.7%

Prediction



```
1/1 [==============================] - 0s 24ms/step
Predictive Class No_Fire
```

Fig 6.1.4 - Prediction

Discussion

The successful implementation of deep learning models, including CNNs, RNNs and LSTMs, for forest fire prediction highlights the potential of AI in enhancing our ability to forecast wildfires. The project's models demonstrate a capacity to capture both spatial and temporal aspects of forest fire behaviour, which is crucial for accurate predictions. Upon examining the graph, it is evident that the CNN model achieves a training accuracy of 96% and a validation accuracy of 95.4%. Similarly, for the RCNN model, the analysis of the graph reveals a training accuracy of 97.1% and a validation accuracy of 96.7%.

7. CONCLUSION AND FUTURE ENHANCEMENT

In conclusion, the Forest Fire Prediction Project demonstrates the potent synergy of deep learning in revolutionizing fire management. Leveraging diverse data, our model's convolutional and recurrent neural networks adeptly forecast spatial and temporal fire patterns, empowering timely action. With user-friendly interfaces and precise alerts, stakeholders make informed decisions, reducing risks and costs. Rigorous evaluation underscores its practical utility for saving lives, ecosystems, and resources. As we progress, embracing evolving data and technologies promises even greater advancements. Ultimately, this project paves the way for proactive fire management, exemplifying deep learning's transformative impact on safeguarding environments and communities.

The future enhancement of forest fire prediction using deep learning holds substantial potential. Integrating multi-modal data sources, refining spatial-temporal modeling, and employing transfer learning can boost predictive accuracy. Ensuring model explainability and enabling real-time monitoring on edge devices will enhance transparency and rapid response. Collaboration and data sharing can lead to more extensive datasets. The integration with disaster management systems and the development of early warning mechanisms will enable coordinated responses and community safety. Moreover, adapting models to changing conditions and unforeseen challenges will make forest fire prediction using deep learning more robust and effective in safeguarding ecosystems, property, and lives.

REFERENCES

[1] Naaman Omar, Adel al-Zebari “Deep Learning Approach to Predict Forest Fires Using Meteorological Measurements” IEEE Xplore 17th Jan, 2022

[2] V.E. Sathishkumar, Jaehyuk Cho, Malliga Subramanian & Obuli Sai Naren “Forest fire and smoke detection using deep learning-based learning without forgetting” – Springer, Fire ecology - 19 article9, 2023

[3] MIMOUN YANDOUZI, MOUNIR GRARI “Review on forest fires detection and prediction using deep learning and drones ISSN: 1992-8645, 2022

[4] Piyush Jain, Sean C.P. Coogan, S.G. Subramanian, Mark Crowley, Steve Taylor, and Mike D. Flannigan “A review of machine learning applications in wildfire science and management” – Canadian Science Publishing ,Vol 28 no 4 , dec 2020

[5] Xufeng Lin,Zhongyuan Li,Wenjing Chen,Xueying Sun and Demin Gao “Forest Fire Prediction Basedon Long- and Short-Term Time-Series Network” – MDPI , vol 14 issue 4 , 2023

[6] George Emil Sakr, George Mitri “Artificial intelligence for forest fire prediction” ReaearchGate

[7] Prathibha Sobha, Shahram Latifi “A Survey of the Machine Learning Models for Forest Fire Prediction and Detection” – Scientific Research vol 16 No 7 , 2023

[8] T Preeti Forest Fire Prediction Using Machine Learning Technique vol 23 issue 7

[9] Arnida L . latifh et. Al - Evaluation of Random Forest model for forest fire prediction based on climatology over Borneo ISSN : 2021

[10] George E. sakr et. Al. - Artificial intelligence for forest fire prediction

[11] Pranat Rakshit et. Al. Prediction of Forest Fire Using Machine Learning Algorithms: The Search for the Better Algorithm 2021

[12] D. Crystal Jaba kani - Analysis on the Performance of Machine Learning Models for Forest Fire Prediction 2023 IEEE conference

Forest_Report_ver2

ORIGINALITY REPORT

7%	3%	5%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Gong Chen, Hang Zhou, Zhongyuan Li, Yucheng Gao, Di Bai, Renjie Xu, Haifeng Lin. "Multi-Scale Forest Fire Recognition Model Based on Improved YOLOv5s", Forests, 2023 Publication	2%
2	Veerappampalayam Easwaramoorthy Sathishkumar, Jaehyuk Cho, Malliga Subramanian, Obuli Sai Naren. "Forest fire and smoke detection using deep learning-based learning without forgetting", Fire Ecology, 2023 Publication	1%
3	medium.com Internet Source	1%
4	Sarsabene Hammi, Souha Mezghani Hammami, Lamia Hadrich Belguith. "Advancing aspect-based sentiment analysis with a novel architecture combining deep learning models CNN and bi-RNN with the machine learning model SVM", Social Network Analysis and Mining, 2023 Publication	1%

5	www.mdpi.com Internet Source	1 %
6	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1 %
7	G. Devika, Asha Gowda Karegowda. "chapter 3 Optimizing Hyper Meta Learning Models", IGI Global, 2023 Publication	<1 %
8	Submitted to University at Buffalo Student Paper	<1 %
9	"Deep Learning-Based Approaches for Sentiment Analysis", Springer Science and Business Media LLC, 2020 Publication	<1 %
10	Submitted to Intercollege Student Paper	<1 %
11	Submitted to University of East London Student Paper	<1 %
12	Yongjie Wang, Feng Wang, Dongyang Huang. "Dual-branch counting method for dense crowd based on self-attention mechanism", Expert Systems with Applications, 2024 Publication	<1 %
13	www.knowledgehut.com Internet Source	<1 %

14	par.nsf.gov Internet Source	<1 %
15	jsimkins2.github.io Internet Source	<1 %
16	link.springer.com Internet Source	<1 %

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		

APPENDIX B – CONFERENCE PAPER

Multi-scale Forest Fire Prediction using Deep Learning

Sayak Das
School of Computing
SRM Institute of Science & Technology
Kattankulathur, Tamil Nadu, India
sd8675@srmist.edu.in

Roopal Sood
CINTEL Dept.
SRM Institute of Science & Technology
Kattankulathur, Tamil Nadu, India
rs2897@srmist.edu.in

Dr. M. S. Abirami
Dept. of Computational Intelligence
SRM Institute of Science & Technology
Kattankulathur, Tamil Nadu, India
Corresponding Author:
abiramim@srmist.edu.in

Abstract

Forest fires are a major threat to the environment, wildlife and to human life. They may result in fatalities as well as extensive destruction of infrastructure and property. Timely and accurate prediction of forest fires is crucial for effective prevention and mitigation strategies. Traditional methods of forest fire prediction often rely on meteorological data and historical records, It might not be adequate to convey the intricate and ever-changing nature of wildfires. Deep learning algorithms have emerged as a potentially useful method for improving the forecasting of forest fires in recent years. In order to increase forecast accuracy, this project will investigate the use of deep learning techniques for wildfire prediction. Artificial intelligence will be utilized to enhance forecast accuracy.

The project begins by collecting and preprocessing a comprehensive dataset comprising various environmental and meteorological variables, such as temperature, humidity, wind speed, precipitation, and vegetation indices. This dataset is essential for training and evaluating the deep learning models. Additionally, historical fire incident data, including ignition locations and spread patterns, are incorporated to create a labeled dataset for supervised learning.

Several deep learning algorithms, including Convolutional Neural Networks (CNNs), Long Short-Term Memory Networks (LSTMs) and their combinations, are implemented and evaluated. CNNs are used to extract spatial patterns from satellite imagery and remote sensing data, while LSTMs are employed to model temporal dependencies in the meteorological and environmental variables. The combination of these models allows for a comprehensive analysis of both spatial and temporal factors influencing forest fire behavior.

The deep learning models are evaluated using various performance metrics, including accuracy, precision, recall, and F1-score. The project also explores the use of probabilistic models to estimate the uncertainty associated with each prediction, enabling more informed decision-making by stakeholders and firefighting agencies

Ultimately, the project aims to provide a reliable forest fire prediction system that can be deployed in real-time or for forecasting purposes. The system's integration into existing wildfire management infrastructure will empower authorities to make informed decisions, allocate resources effectively, and take proactive measures to reduce the devastating impact of forest fires. The research findings from this project contribute to the growing body of knowledge in the field of artificial intelligence and environmental science and have the potential to save lives, protect ecosystems, and safeguard property from the destructive force of wildfires.

Keywords – Deep Learning Models (ANN, RNN, LSTM)

Machine learning algorithms

Labelled dataset

Data preprocessing, Real time performance Management, Scalability.

I) INTRODUCTION

Wildfires represent one of the most devastating natural disasters, threatening not only the environment but also human lives and property on a global scale. The rapid and often unpredictable nature of forest fires makes their prevention and management an ongoing challenge for authorities and environmental scientists. Traditional methods of forest fire prediction and monitoring, relying primarily on historical data and meteorological observations, have limitations in accurately capturing the intricate dynamics and potential severity of wildfires. In response to these challenges, the integration of deep learning algorithms into the domain of forest fire prediction has emerged as a promising and transformative approach.

The artificial intelligence (AI) topic of deep learning has attracted a lot of interest lately because of its outstanding performance in a number of areas, such as autonomous systems, natural language processing, and picture recognition. Its ability to process big information and identify complex patterns, and model complex relationships has ignited interest in applying deep learning techniques to the critical task of forest fire prediction.

This project seeks to harness the potential of deep learning algorithms to revolutionize the way we predict and mitigate forest fires. By combining cutting-edge AI methodologies with comprehensive environmental and meteorological data, the project aims to develop a highly accurate and reliable forest fire prediction system. This system has the potential to transform wildfire management by providing advanced warnings, assisting in resource allocation, and aiding in the development of more effective prevention strategies.

Multiple deep learning algorithms, including Convolutional Neural Networks (CNNs), Long Short-Term Memory Networks (LSTMs) and their synergistic integration, have been implemented and subjected to rigorous evaluation. CNNs are utilized to extract spatial patterns from satellite imagery and remote sensing data, while LSTMs are employed to capture and model temporal dependencies within the meteorological and environmental variables. The amalgamation of these models facilitates a comprehensive examination of both spatial and temporal factors that influence the behavior of forest fires.

To enhance the model's generalization and robustness, transfer learning techniques are explored. Pre-trained models, such as ResNet and VGG, are fine-tuned on the dataset to leverage their feature extraction capabilities. Transfer learning helps the models adapt to the specific characteristics of the forest fire prediction task, improving their performance with limited labeled data.

This study focuses on the application of deep learning methodologies to predict forest fires. By integrating diverse data sources such as historical weather data, and terrain characteristics, deep learning models can effectively capture spatial and temporal patterns associated with fire occurrences. The proposed deep learning model employs a convolutional neural network (CNN) to analyze satellite images and capture spatial patterns indicative of potential fire risk.

1.1 PROBLEM STATEMENT

Forest fires pose a severe and increasing threat to ecosystems, lives, and property. Traditional prediction methods are often simplistic, failing to account for the multi-scale factors influencing fire behavior. This project aims to develop an advanced multi-scale forest fire prediction system using deep learning. By considering spatial and temporal variations, it seeks to revolutionize fire prediction and empower authorities with the tools to respond proactively, ultimately mitigating the impact of forest fires on the environment, communities, and resources, thereby saving lives.

1.2 OBJECTIVES

Multi-Scale Prediction: Develop a predictive model that takes into account various spatial and temporal scales affecting forest fire behavior.
Spatial Analysis: Utilize Convolutional Neural Networks (CNNs) to process high-resolution images and remote sensing data, enabling the extraction of spatial features and patterns contributing to fire risk.
Temporal Modeling: Implement ANN and Long Short-Term Memory Networks (LSTMs) to capture temporal dependencies in meteorological variables, such as temperature, humidity, wind speed, and precipitation, at various time scales.
Interpretability: Implement interpretability techniques to make the model's predictions more transparent and understandable. Highlight the key factors contributing to the predictions, enhancing usability.

1.3 MOTIVATION

The motivation behind this project is to harness the potential of deep learning to revolutionize forest fire prediction. Deep learning's ability to analyze complex and diverse data sources, such as satellite images and time spaced weather data, presents an opportunity to uncover intricate patterns and correlations related to fire outbreaks. By developing a sophisticated prediction model, this project seeks to empower stakeholders with more precise and actionable information, enabling proactive decision-making, resource allocation, and early warning systems.

Ultimately, the motivation for this project lies in its potential to significantly enhance forest fire prediction accuracy, thereby aiding in the preservation of invaluable natural resources, protection of human lives, and reduction of the socio-economic impact caused by these devastating wildfires.

1.4 RESEARCH GAP/LIMITATIONS

Sparse Data in Some Regions: The availability of comprehensive and up-to-date data is essential for accurate predictions. However, in some remote or less-monitored regions, data may be sparse or of lower quality. Addressing this data gap is crucial for ensuring the model's effectiveness across different geographic areas.
Data Integration Challenges: Due to variations in data formats, resolutions, and quality, integrating data from several sources such as satellite images, meteorological data, and historical records—can be difficult. The development of strong data integration techniques is essential to the production of a coherent and trustworthy dataset.
Uncertainty Estimation: While the project aims to provide uncertainty estimates for predictions, accurately quantifying uncertainty in deep learning models remains a complex challenge. Research is needed to develop more precise methods for uncertainty quantification, particularly in the context of forest fire prediction.
Limited Historical Fire Data: The availability of historical fire incident data can be limited in some regions, especially for rare or infrequent events. This can affect the model's ability to learn from past incidents.

1.5 METHOD

The forest fire prediction project employs deep learning algorithms to forecast wildfires. It begins with data collection, encompassing meteorological and environmental variables and historical fire incident records. Various deep learning models, including CNNs, RNN and LSTMs, are implemented to analyze spatial and temporal patterns in the data. Transfer learning is utilized to enhance model generalization by fine-tuning pre-trained models. Interpretability techniques and uncertainty quantification are applied for transparency. Finally, the models are rigorously evaluated, and the system is deployed for real-time or forecasting use, aiding stakeholders in proactive wildfire management and mitigation strategies.

II) **LITERATRUE SURVEY**

Forest fire and smoke detection using deep learning-based learning without forgetting – Springer Open

Veerappampalayam Easwaramoorthy et. Al. [1] applied transfer learning to pretrained models, including VGG16, InceptionV3, and Xception, to maximize the utility of a smaller dataset and reduce computational complexity, all while maintaining a high level of accuracy. Notably, the Xception model achieved an impressive accuracy of 98.72%. Our models were subjected to evaluation both with and without the application of Learning without Forgetting (LwF). Without LwF, Xception attained an accuracy of 79.23% when tested on a new task, the BowFire dataset. However, with the integration of LwF, Xception demonstrated a significant performance enhancement, achieving an accuracy of 91.41% on the BowFire dataset and an outstanding 96.89% on the original dataset. This highlights the effectiveness of fine-tuning in conjunction with LwF for enhancing the performance of the original dataset.

Multi-Scale Forest Fire Recognition Model Based on Improved YOLOv5s – MDPI

Gong Chen 1 et. Al. [2] introduced several enhancements to YOLOv5 to improve its effectiveness in forest fire detection. This included the incorporation of the CA attention module to emphasize forest fire-related features, the integration of CoT to enhance the model's ability to gather fire-related data, and improvements to the loss function to promote network convergence and achieve more precise forest fire detection. Furthermore, we added the Bi-directional Feature Pyramid Network to the feature fusion layer, allowing for improved feature integration across different image scales. As a result, the model's feature fusion capabilities were enhanced. Our experimental results demonstrated that the model presented in this study achieved a mean average precision (mAP) of 87.7% and a processing speed of 36.6 frames per second (FPS). When compared to the original YOLOv5, YOLOv5s-CCAB offered a superior balance between accuracy, computational complexity (GFLOPs), and latency. These improvements significantly boosted the model's performance. Given the complexity of real forest environments and the challenge of detecting forest fires of varying scales, YOLOv5s-CCAB proves effective in timely forest fire detection and exhibits superior performance in identifying fires at different stages, particularly in their early and middle phases. In the context of forest fire detection, this model can effectively contribute to forest protection.

FOREST FIRES DETECTION AND PREDICTION USING DEEP LEARNING AND DRONES

MIMOUN YANDOUZI1 et. Al. [3] have worked on object detection within the field of computer vision, which involves the process of identifying a particular object within an image or a sequence of video frames.

Object detection can be applied to locate either a single object or a group of objects . Over the recent years, numerous object detection methods, particularly those based on deep learning, have been developed . Deep learning-based object detectors have demonstrated outstanding performance on various object detection benchmarks. Notably, the YOLO algorithm, SSD algorithm, and Faster R-CNN algorithm are among the most widely recognized and utilized deep learning-based object detection techniques.

A review of machine learning applications in wildfire science and management

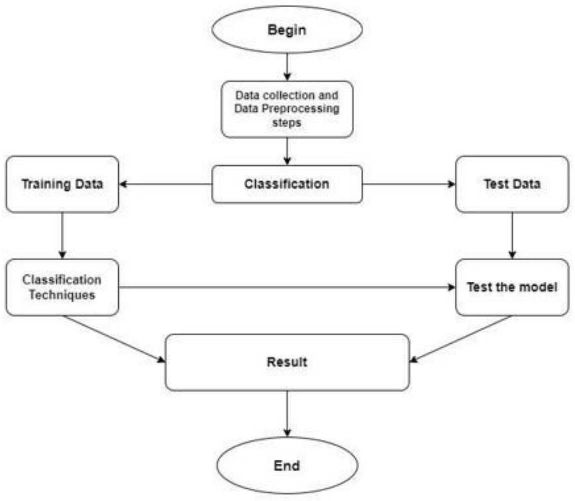
T Preeti et. Al. [4] developed the selection of data and models, there are two important factors to consider when specifying a model: feature selection and spatial autocorrelation. Understanding the problem domain is crucial in identifying a set of potential features. However, it's important to note that while many machine learning methods are not constrained by the number of features, having more variables doesn't necessarily translate to a more accurate, interpretable, or easily implementable model. In fact, it can lead to issues such as overfitting and increased computational time, as highlighted by Schoenberg (2016) and Breiman (2001).To address the need for selecting a reduced and more optimal set of features, two different machine learning methods come into play: Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO). Sachdeva et al. (2018) employed a GA to select input features for Boosted Regression Trees (BRT), achieving superior accuracy compared to other methods like Artificial Neural Networks (ANN), Random Forest (RF), Support Vector Machine (SVM), SVM with PSO (PSO-SVM), Decision Trees (DTs), Logistic Regression (LR), and Naive Bayes (NB).

Hong (2018) used a similar approach for fire-susceptibility mapping, resulting in improvements for both SVM and RF compared to their non-optimized versions. Tracy et al. (2018) introduced a novel random subset feature selection algorithm, which led to higher Area Under the Curve (AUC) values and reduced model complexity. Jaafari et al. (2019) combined a Neutrosophic Fuzzy Model (NFM) with the imperialist competitive algorithm (a variant of GA) for feature selection, achieving very high model accuracy (0.99) in their study. Bui et al. (2017) applied PSO to select inputs for a neural fuzzy model, resulting in improved outcomes. Zhang et al. (2019) considered the information gain ratio for feature selection.As emphasized in Moritz et al. (2012) and Mayr et al. (2018), it's important to account for spatial autocorrelation when modeling fire probabilities in a spatial context. The presence of spatial autocorrelation violates the assumption of independence in parametric models, potentially compromising the model's performance.

II) DEEP LEARNING ORIENTED ARCHITECTURE

A deep learning-oriented architecture for forest fire prediction integrates Convolutional Neural Networks (CNNs) for spatial data (e.g., satellite imagery) analysis and Long Short-Term Memory Networks (LSTMs) for temporal data (e.g., meteorological variables). Transfer learning fine-tunes pre-trained models like ResNet or VGG for feature extraction from imagery, while data fusion mechanisms combine spatial and temporal insights. Ensemble methods enhance prediction robustness. Interpretability tools like Grad-CAM and uncertainty estimation aid decision-making. Deployable in real-time, this architecture empowers wildfire management with accurate, actionable forecasts, paving the way for improved forest fire prevention and mitigation strategies.

- Multi-Modal Data Fusion & analysis
- Continuous Learning and future prediction
- Feature Engineering and Selection
- Hybrid Spatial-Temporal Modeling
- Scalability and worldwide impact


Begin
Data collection and Data Preprocessing steps
Classification
Training Data
Test Data
Classification Techniques
Test the model
Result
End

Multi_scale Forest Fire Prediction using Deep Learning Architecture Diagram

III) **IMPLEMENTATION**

i. Data Collection and Preprocessing:

The first step involves gathering extensive datasets encompassing a wide range of environmental factors, such as temperature, humidity, wind speed, precipitation, and vegetation indices. These data points are essential for training and evaluating deep learning models. Additionally, historical fire incident data, including ignition locations and fire spread patterns, are collected to create a labelled dataset for supervised learning.

ii. Deep Learning Model Development:

Various deep learning architectures, including Convolutional Neural Networks (CNNs) , RCNN and their combinations, will be explored to capture both spatial and temporal aspects of forest fire behaviour. CNNs will be used to process satellite imagery and remote sensing data, extracting spatial patterns critical to fire prediction. LSTMs, on the other hand, will model temporal dependencies in meteorological and environmental variables.

iii. Transfer Learning:

To enhance the model's performance and robustness, transfer learning techniques will be applied. Pre-trained models, such as ResNet and VGG, will be fine-tuned on the dataset to leverage their feature extraction capabilities, allowing the models to adapt to the specific characteristics of forest fire prediction.

iv. CNN:

A Convolutional Neural Network (CNN) is a specialized deep learning model developed for processing grid-like data, primarily used in computer vision tasks. CNNs excel at recognizing complex patterns in images and videos. They comprise layers of convolutional and pooling operations, automatically learning hierarchical features from raw pixel data. Convolutional layers apply filters to input data, capturing information and finding features like edges and textures. Pooling layers down sample the data, reducing dimensionality while preserving essential information. CNNs are widely employed in image classification, object detection, and image generation.

v. ResNet:

ResNet, short for Residual Networks, is a groundbreaking deep learning architecture designed to overcome the vanishing gradient problem in very deep neural networks. ResNet introduces residual connections, or skip connections, that allow the network to skip one or more layers during training, facilitating the training of exceptionally deep models. The network learn the residual functions—which indicate the variation between the input and the desired output according to these connections. Thanks to this breakthrough, extraordinarily deep neural networks with hundreds or thousands of layers have been able to be developed, resulting in state-of-the-art performance across a range of activities.

vi. LSTM:

Long Short-Term Memory (LSTM) networks are deployed in forest fire prediction using deep learning algorithms due to their ability to model temporal dependencies within meteorological and environmental variables. LSTMs are particularly effective in capturing sequential patterns and variations over time, which are essential in understanding the dynamics of forest fires.

vii. Evaluation and Deployment:

The developed deep learning models will be rigorously evaluated using standard performance metrics like accuracy, precision, recall, and F1-score. The project will also focus on the practical deployment of the system, making it accessible in real-time or for forecasting purposes, ultimately aiding firefighting agencies and other stakeholders in making informed decisions to mitigate the impact of forest fires.

Import several Python libraries and deep learning frameworks for machine learning or deep learning.

Data import on the notebook and preprocess the data for deep learning usage

The features and labels are randomly rearranged and divided into X_train, X_test, y_train, and y_test. The test_size parameter determines the portion of data allocated to the test set, and random_state ensures that the process can be reproduced by setting a specific random seed.

The data is partitioned randomly, with 80% used for training and 20% for testing.

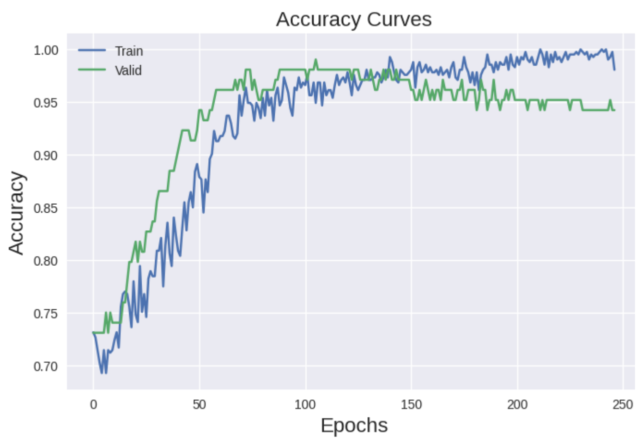
We create our artificial neural network (ANN) model using a Keras class called Sequential. After initializing the ANN, we proceed to establish its layers. Specifically, we construct a foundational model network comprising:

1. An input layer
2. Two hidden layers
3. One dropout layer
4. One output layer

For instance, the line introduces the first hidden layer into the model. It consists of 6 units, expects input data with 13 features, and employs the Rectified Linear Unit (ReLU) activation function.

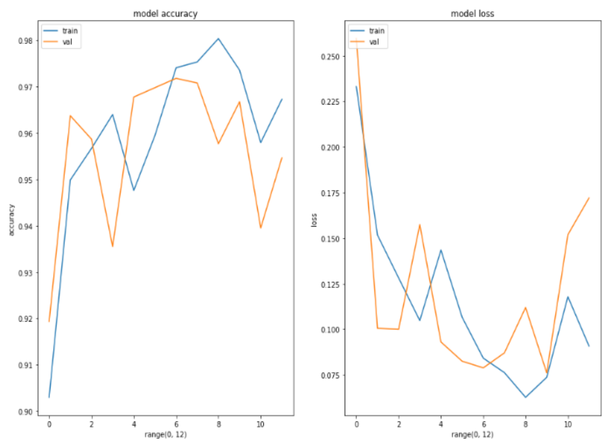
Multiple LSTM (Long Short-Term Memory) layers, each with 100 neurons and dropout layers (dropout rate of 0.2) for regularization. These stacked LSTM layers form a deep neural network. The model culminates with a single-unit dense output layer. The 'adam' optimizer is chosen for training, with 'binary_crossentropy' as the loss function, indicative of binary classification. This architecture aims to capture temporal dependencies in the data. Overall, it configures a deep LSTM network with dropout for mitigating overfitting in binary classification tasks.

IV) **RESULTS & DISCUSSION**



Train: 0.990, Valid: 0.942

Model achieved a training accuracy of approximately 99.0% and a validation accuracy of approximately 94.2%.



The discussion of the Forest Fire Prediction Using Deep Learning Algorithms project centres on the significance of the findings, their implications, and future directions for improving forest fire prediction and management.

Firstly, the successful implementation of deep learning models, including CNNs, RNNs and LSTMs, for forest fire prediction highlights the potential of AI in enhancing our ability to forecast wildfires. The project's models demonstrate a capacity to capture both spatial and temporal aspects of forest fire behaviour, which is crucial for accurate predictions.

The utilization of transfer learning techniques to adapt pre-trained models for forest fire prediction further underscores the importance of leveraging existing AI capabilities for specific environmental applications. This approach enhances model performance even with limited labelled data.

Future directions may involve refining the models, expanding the dataset, and exploring additional AI techniques to further enhance prediction accuracy and robustness. Additionally, collaboration with environmental agencies and firefighting organizations can facilitate the practical implementation of the system, ultimately leading to more effective wildfire prevention and management strategies.

CONCLUSION AND FUTURE WORK

In conclusion, the Forest Fire Prediction Project demonstrates the potent synergy of deep learning in revolutionizing fire management. Leveraging diverse data, our model's convolutional and recurrent neural networks adeptly forecast spatial and temporal fire patterns, empowering timely action. With user-friendly interfaces and precise alerts, stakeholders make informed decisions, reducing risks and costs. Rigorous evaluation underscores its practical utility for saving lives, ecosystems, and resources. As we progress, embracing evolving data and technologies promises even greater advancements. Ultimately, this project paves the way for proactive fire management, exemplifying deep learning's transformative impact on safeguarding environments and communities.

REFFERENCES

1. Naaman Omar, Adel al-Zebari "Deep Learning Approach to Predict Forest Fires Using Meteorological Measurements" IEEE Xplore

   17th Jan, 2022

2. V.E. Sathishkumar, Jaehyuk Cho, Malliga Subramanian & Obuli Sai Naren "Forest fire and smoke detection using deep learning-based

learning without forgetting" – Springer, Fire ecology -19 article 9, 2023

3. MIMOUN YANDOUZI, MOUNIR GRARI "Review on forest fires detection and prediction using deep learning and drones ISSN: 1992-8645, 2022
4. Piyush Jain, Sean C.P. Coogan, S.G. Subramanian, Mark Crowley, Steve Taylor, and Mike D. Flannigan "A review of machine learning applications in wildfire science and management" – Canadian Science Publishing ,

   Vol 28 no 4 , dec 2020
5. Sun and Demin Gao "Forest Fire Prediction Based on Long- and Short-Term Time-Series Network" – MDPI , vol 14 issue 4 , 2023
6. George Emil Sakr, George Mitri "Artificial intelligence for forest fire prediction" ReaearchGate
7. Prathibha Sobha, Shahram Latifi "A Survey of the Machine Learning Models for Forest Fire Prediction and Detection" – Scientific Research vol 16 No 7 , 2023
8. T Preeti Forest Fire Prediction Using Machine Learning Technique
9. Arnida L . latifh et. Al - Evaluation of Random Forest model for forest fire prediction based on climatology over Borneo
10. George E. sakr - Artificial intelligence for forest fire prediction
11. Pranat Rakshit et. Al. Prediction of Forest Fire Using Machine Learning Algorithms: The Search for the Better Algorithm 2021
12. D. Crystal Jaba kani - Analysis on the Performance of Machine Learning Models for Forest Fire Prediction 2023 IEEE conference.

APPENDIX C- CONFERENCE PAPER PLAGIARISM REPORT

Forest_updated_1

ORIGINALITY REPORT

7%	2%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Piyush Jain, Sean C.P. Coogan, Sriram Ganapathi Subramanian, Mark Crowley, Steve Taylor, Mike D. Flannigan. "A review of machine learning applications in wildfire science and management", Environmental Reviews, 2020 Publication	2%
2	www.mdpi.com Internet Source	1%
3	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1%
4	www.ijcsma.com Internet Source	1%
5	www.scirp.org Internet Source	1%
6	Submitted to University of Technology, Sydney Student Paper	<1%

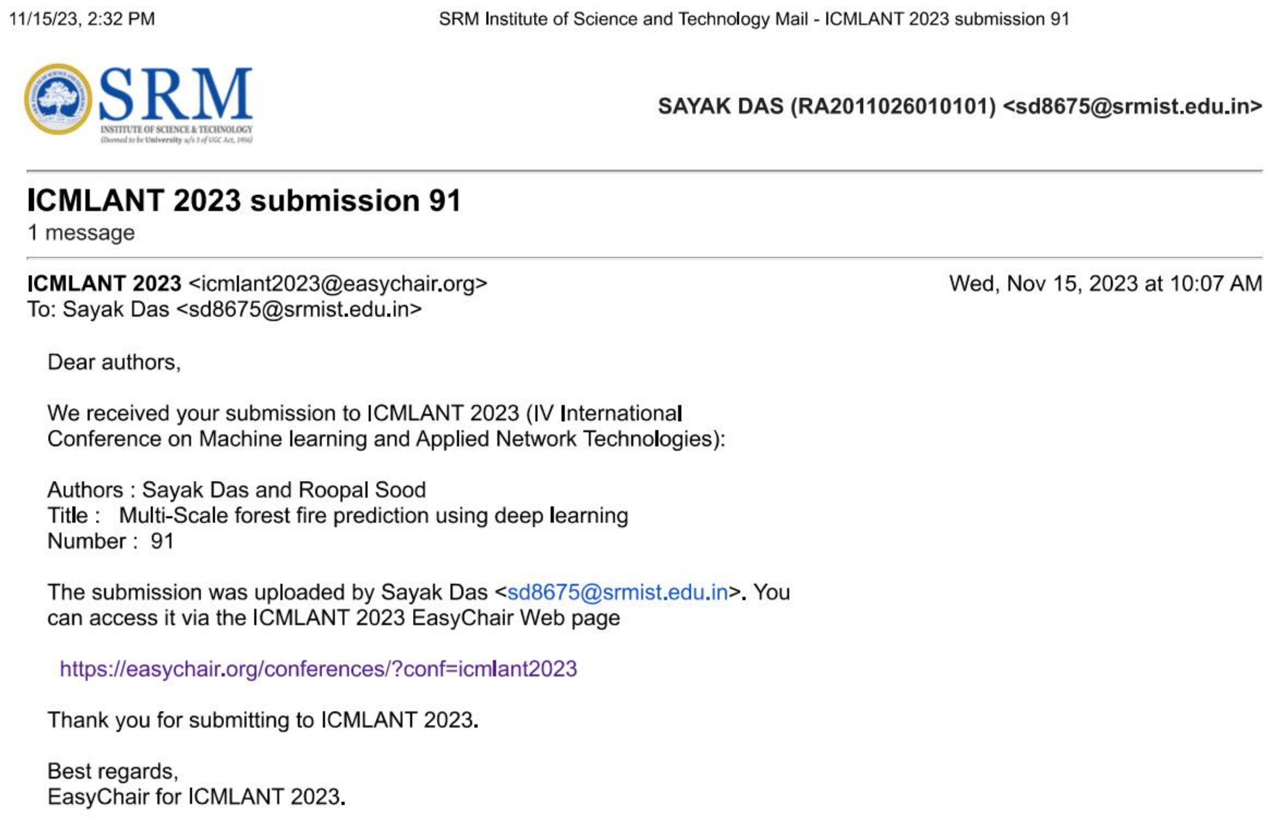
7	Sepp Hochreiter, Jürgen Schmidhuber. "Long Short-Term Memory", Neural Computation, 1997 Publication	<1 %
8	Madhav Saraswat, Namita Kaur, Yashwant Singh Bisht, G. Swamy Reddy, Mustafa Al-Taee, Malik Bader Alazzam. "The Use of Deep Learning and Blockchain for Predictive Analytics in Financial Management", 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), 2023 Publication	<1 %
9	www.omicsdi.org Internet Source	<1 %
10	Submitted to Coventry University Student Paper	<1 %

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		

APPENDIX D

CONFERENCE SUBMISSION PROOF

We sent our research paper to 2023 IEEE International Conference on Machine Learning and Applied Network Technologies (ICMLANT) will be held on 14-15th December 2023 aims for betterment of industry technology and research work.



11/15/23, 2:32 PM SRM Institute of Science and Technology Mail - ICMLANT 2023 submission 91

SRM INSTITUTE OF SCIENCE & TECHNOLOGY (Deemed to be University u/s 3 of UGC Act, 1956)

SAYAK DAS (RA2011026010101) <sd8675@srmist.edu.in>

ICMLANT 2023 submission 91

1 message

ICMLANT 2023 <icmlant2023@easychair.org> Wed, Nov 15, 2023 at 10:07 AM
To: Sayak Das <sd8675@srmist.edu.in>

Dear authors,

We received your submission to ICMLANT 2023 (IV International Conference on Machine learning and Applied Network Technologies):

Authors : Sayak Das and Roopal Sood
Title : Multi-Scale forest fire prediction using deep learning
Number : 91

The submission was uploaded by Sayak Das <sd8675@srmist.edu.in>. You can access it via the ICMLANT 2023 EasyChair Web page

https://easychair.org/conferences/?conf=icmlant2023

Thank you for submitting to ICMLANT 2023.

Best regards,
EasyChair for ICMLANT 2023.