



National
College of
Ireland

WILDFIRE PRECISION USING MACHINE LEARNING

MSc Research Project
MSc Data Analytics

JAI MAHESH CHAUHAN
StudentID: X20208375

School of Computing
National College of Ireland

Supervisor: Rejwanul Haque

**National College of Ireland
Project Submission Sheet
School of Computing**



Student Name:	JAI MAHESH CHAUHAN
Student ID:	X20208375
Programme:	MSc Data Analytics
Year:	2022-2023
Module:	MSc Research Project
Supervisor:	Rejwanul Haque
Submission Due Date:	14/08/2023
Project Title:	Wildfire Prediction Using Machine Learning
Word Count:	8424
Page Count:	25
Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	JAI MAHESH CHAUHAN
Date:	14th August 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Wildfire Prediction Using Machine Learning

Jai Chauhan

x20208375

1 Abstract

This report discusses wildfire research methods, data collection, and analysis. This study covers ecological, meteorological, and economic aspects of wildfire studies. It helps students plan, perform, and comprehend research assignments. The guidebook emphasizes cross-disciplinary collaboration in environmental studies, geography, and computer science. It encourages wildfire awareness and imaginative solutions. Finally, the report advances wildfire research significantly. By providing a comprehensive pathway, learners can make significant contributions to wildfire reduction and control, advancing the field. Students can improve wildfire reduction and control with this comprehensive guide. Its systematic approach prepares a new generation of scholars to tackle wildfires' complex challenges.

Keywords - Wildfire Prediction, Tuning Hyperparameter, Machine learning, Feature Engineering, Model Selection, CNN

Table of Contents

1	Abstract	1
2	Introduction.....	3
2.1	Research Question	3
2.2	Research Hypothesis.....	3
3	Literature Review	3
3.1	Wildfire Prediction Modelling Using Convolutional Neural Networks (CNNs).....	4
3.2	Data Understanding Techniques for Wildfire Prediction	4
3.3	Machine Learning Methods for Wildfire Prediction	5
3.4	Table	6
4	Research Methodology	7
4.1	Data Collection	8
4.2	Data Exploration.....	8
4.3	Data preprocessing.....	8
4.4	Feature Engineering.....	8
4.5	Data Integration.....	8
4.6	Data Sampling.....	8
4.7	Data Analysis.....	9
5	Design Specification.....	12
5.1	Preparing Data	13
5.2	Effect on Model Execution	15
6	Implementation.....	16
6.1	Cleaning Up the Dataset.....	16
6.2	Discretionary Datasets	16
6.3	Sequential CNN Model Design.....	17
6.4	The Sequential CNN Model Training.....	18
6.5	Tuning of Hyperparameters.....	19
6.6	Visualizations.....	19
6.7	Model Assessment	20
7	Evaluation.....	21
7.1	Experiment: Historical Data Analysis and Feature Engineering.....	21
7.2	Experiment: Model Optimization and Hyperparameter Tuning.....	22
7.3	Experiment: Real-time Wildfire Prediction and Monitoring.....	22
8	Conclusion and Future Work.....	23
9	References	23

2 Introduction

This report analyses the wildfire prediction modeling and analyses the wildfire prediction dataset. In the 1st Section, data understanding is understood the data like the shape of the data and metadata information of the dataset. In the data preparation section, prepare the data by loading the data for further processes like reading the image dataset. In the data analysis section, analyze the dataset by performing the operations on the dataset. In the data modeling section, build the model using the Convolutional Neural Network (CNN). And at the last train the model and check the accuracy of the model. Check the accuracy by plotting the histogram plot between the epochs and accuracy. Because wildfires are becoming more dangerous than ever all across the world, wildfire forecast has grown in importance. The availability of thorough and trustworthy datasets is essential for making precise predictions about the occurrence and behavior of wildfires. These statistics offer useful details on numerous environmental elements that affect how wildfires start and spread. To develop precise predictive models, wildfire prediction dataset modeling looks for patterns, correlations, and trends within the data. To analyze the datasets and create reliable predictive models, machine learning methods and deep learning methods neural networks are frequently used. These models predict the likelihood and behavior of wildfires.

2.1 Research Question

How can we use machine learning and data analysis to accurately predict and model the spread of wildfires? The research revolves around a single guiding question: How can we effectively design and train a Convolutional Neural Network (CNN) model to predict wildfires with precision?

2.2 Research Hypothesis

The objective of this study was to develop a more accurate, generalizable, and approachable model for predicting wildfires by applying cutting-edge machine-learning techniques and meticulous data analysis. This model has the potential to considerably enhance firefighting strategies, resource allocation, and the comprehension of the human and environmental costs of wildfires.

3 Literature Review

Wildfires have arisen as a significant ecological test, presenting huge dangers to environments, living souls, and property around the world. Because of their rising risk, wildfire forecasts have acquired vital significance in the space of ecological exploration and fiasco the board. To resolve this basic issue, this writing audit dives into the domain of wildfire forecast displaying and information figuring out procedures. The essential center is to investigate the use of AI, Machine Learning, and deep learning methods, especially Convolutional Neural Networks (CNNs), in precisely anticipating wildfire events and grasping their way of behaving. In addition, this report plans to stress the imperative job of solid and reliable datasets in refining prescient models. By analyzing the

existing reviews, one can look to reveal insight into the different systems utilized in information assortment, investigation, preprocessing, highlight designing, and information perception, all of which add to fathoming wildfire datasets. Through this comprehensive examination, one tries to give important experiences into the present status of wildfire expectation research and recognize possible roads for additional headways in this essential field.

3.1 Wildfire Prediction Modelling Using Convolutional Neural Networks (CNNs)

(Marjani & Mesgari, 2023) examine CNNs' ability to predict wildfire outbreaks and behavior. Wildfires have cost the climate billions in the past twenty years. Wildfire spread must be predicted with an accurate model. In this review, a multi-kernel convolution neural network (CNN) profound learning model was proposed to predict US wildfire spread based on elevation, wind direction and speed, minimum and maximum temperatures, humidity, precipitation, drought index, Normalized difference vegetation index (NDVI), and energy discharge part. Multipart CNN can predict pixel fires. The multi-part CNN model outperformed other models in precision and F1 score. The test informative index predicted more accurate results with CNNs without a multi-piece component and fixed section size. The "multi-kernel CNN" model scored 98.6 general exactness and 70.97 F1 on test data. The multi-kernel CNN model can predict wildfire spread. The model generates binary values for each pixel-based on a $64 \times 64 \times 12$ tensor input. 12 groups include elevation, wind direction and speed, minimum, maximum temperatures, humidity, precipitation, drought index, NDVI, and energy discharge. New wildfire spread statistics yielded intriguing results. The multi-kernel CNN model outperformed the wildfire dataset distributor. Multi-kernel CNN can also extract higher-level highlights and aid in training wildfire spread designs is shown in Marjani & Mesgari, 2023. Multichannel CNN networks use complex parameters to separate data. This may provide us with new ideas on how to best manage large data and obtain vital data. When cycles over a step, the multi-bit system has greatly increased the model's exhibition. Third, using the test dataset, the proposed model outperforms a CNN model without a multi-kernel instrument and fixed kernel size. Up-sampling or convolution transpose layers can also simplify models.

3.2 Data Understanding Techniques for Wildfire Prediction

According to (Perez-Porras et al., 2021), this paper examines information collection, investigation, preprocessing, and highlights design for wildfire expectation datasets. Wildfires are becoming increasingly frequent worldwide, and predicting when and where they will occur is difficult. Starting attack planning requires identifying wildfire events with a high likelihood of becoming large. Wildfire forecast uses material physics-based, statistical, and "Machine Learning" (ML) methods. The dataset for an ML model is imbalanced since the number of wildfires is far higher than the number of large wildfires, resulting in over-fitting or under-fitting. This original edition proposes creating fabricated knowledge from factors of interest and ML models to predict massive wildfires. Five manufactured information age techniques are evaluated using four ML strategies. In "Decision Support Systems (DSS)" for wildfire management, using produced information

improves forecast power. Wildfires occasionally behave like the variables in the review region. These variables were used: Burn area mask: paired raster inclusion where 1 is assigned to every pixel affected by a wildfire and 0 to those affected by a poor person. "Normalized difference vegetation index" "(NDVI)" (Freden et al., 1974): calculated from the nearest Landsat picture for all consumption regions, obtained from the Google Earth Motor. Land Surface Temperature: from the closest MOD11A1.006. (Hudak et al., 2007), from the MODIS program, for all consume regions, from Google Earth Motor. gasoline model: Mediterranean-themed gasoline display. Danger index: predicts wildfire propagation. (Perez-Porras et al., 2021). Fuel model risk index: recalls data for woodland fuel designs and places without timberland vegetation but with agrarian harvests vulnerable to fire spread. Watershed fuel model risk: watershed-scale fuel model risk file mean value. Geographical risk: evaluated by incline, yearly long insolation, and waste thickness. Watershed surface risk: average topographical danger. Local risk index: includes metropolitan regions, transport organizations, authentic legacy, electrical wires, and fuel pipelines that are vulnerable to wildfire. At the watershed scale, the meteorological risk index reveals meteorological factors that will affect wildfire spread. Water stress risk index: measures vegetation humidity based on precipitation and evapotranspiration. Historical risk index: displays wildfire likelihood as part of verifiable recurrence. Slope risk index: examines how slope affects wildfire behavior when fuel coherence rises. Forest vulnerability risk: assessing backwoods habitats and evaluating coherence to increase tree inclusion. A wildfire weather station estimates wind speed, wind direction, relative humidity, and mean temperature continually. Soil moisture: water stress. Consume region expectation research uses somewhat innovative ML methods. These methods estimate the total region consumed and fire incidents until now. However, these results don't account for natural conditions during wildfire start.

3.3 Machine Learning Methods for Wildfire Prediction

As shown in Ghorbanzadeh et al., 2019, the study examines the usage of an ANN as a reliable machine-learning technique for wildfire prediction. ANNs mimic human brain neurons and synapses to find complex data connections. In wildfire forecasts, ANNs can capture complex relationships between topography, weather, and anthropogenic influences. The ANN can study and learn from large data sets, making it a strong option for modeling multidimensional wildfire vulnerability. SVMs also predict wildfires. SVM excels with nonlinear data. This matches the complicated relationships that cause wildfires. SVM creates hyperplanes to separate data kinds. SVM improves wildfire sensitivity mapping by detecting minor correlations between components. The report stresses random forest models for wildfire prediction. RF ensemble learning generates many decision trees and integrates their findings. This approach minimizes overfitting on large and complex datasets. For wildfire prediction, RF simultaneously examines many conditioning elements, revealing their cumulative effect on susceptibility (Ghorbanzadeh et al., 2019). RF feature significance scores help identify wildfire risk characteristics. K-fold cross-validation strengthens "Machine Learning" systems. This divides the wildfire monitoring dataset into k subsets. Various subset combinations construct and verify the machine-learning model. k-fold CV reduces the volatility of testing and training data choices, boosting model generalizability and dependability. The report emphasizes MODIS thermal anomalies data. The report's use of "Machine Learning" methods in the wildfire forecasts framework shows its commitment to using cutting-edge technology to

address wildfire challenges. The report's use of ANN, SVM, RF, k-fold CV, and MODIS data shows its comprehensive approach to improving wildfire forecasting and control.

3.4 Table

TITLE	AUTHOR	ACCURACY	MODEL
A Google Earth Engine Approach for Wildfire Susceptibility Prediction Fusion with Remote Sensing Data of Different Spatial Resolutions	Sepideh Tavakkoli Piralilou	97.5	Support Vector Machine
Spatially-Explicit Prediction of Wildfire Burn Probability Using Remotely-Sensed and Ancillary Data	Chen Shang	92.5	Time Series
Predictive modeling of wildfires: A new dataset and machine learning approach	Younes Oulad Sayad	98.32	ANN
Study of Driving Factors Using Machine Learning to Determine the Effect of Topography, Climate, and Fuel on Wildfires in Pakistan	Warda Rafaqat	85.19	Random Forest
Modeling Forest Fire Spread Using Machine Learning-Based Cellular Automata in a GIS Environment	Yiqing Xu	Better Results	Better Result
Data-Driven Wildfire Risk Prediction in Northern California	Ashima Malik	92	Random Forest
Forest Fire Prediction Based on Long - and Short-Term Time-Series Network	Xufeng Lin	94	LSTNet
THE LARGE-SCALE WILDFIRE SPREAD PREDICTION USING A MULTI-KERNEL	M. Marjani	98.6	CNN

CONVOLUTIONAL NEURAL NETWORK			
Machine Learning Methods and Synthetic Data Generation to Predict Large Wildfires	Fernando-Juan Pérez-Porras	95	RF and SVC
Evaluation of Different Machine Learning Methods and Deep-Learning Convolutional Neural Networks for Landslide Detection	Omid Ghorbanzadeh	87.8	CNN

4 Research Methodology

In the discipline of machine learning, interpreting the data is essential since it paves the way for model construction and subsequent analysis. Understanding the type, structure, and quality of data is necessary for this process, as is spotting trends, linkages, and potential biases in the dataset. Exploring this key feature of machine learning will enable us to gain insightful knowledge, make wise choices, and make sure our models are accurate and reliable. Understanding data is crucial since it forms the basis of all successful machine-learning efforts. It enables academics and data scientists to learn a great deal about the dataset they are using. They can choose the best model and evaluate it by carefully comprehending the data and making informed decisions about feature engineering, model selection, and evaluation. A thorough understanding of the data enables the development of reliable and accurate models by enabling the discovery of hidden patterns, the identification of outliers, and the detection of potential biases (Shang et al., 2020).

Important Elements of Data Understanding

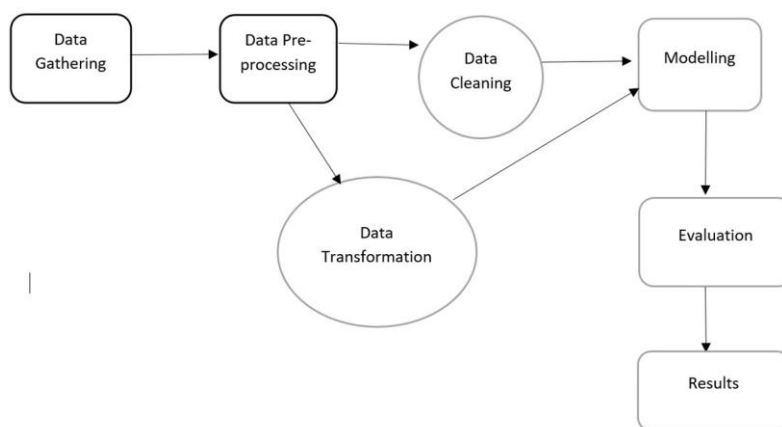


Figure 1: Method

4.1 Data Collection

Gathering essential data from multiple sources, such as databases, APIs, or data providers, is the first step in interpreting data. For a thorough analysis, the data must be complete, representative, and objective.

4.2 Data Exploration

In this stage, the dataset is examined to get a basic grasp of its composition, variables, and distributions. To glean insights and spot possible problems, descriptive statistics, data visualization methods, and exploratory data analysis (EDA) are frequently used.

4.3 Data preprocessing

This procedure deals with outliers, missing numbers, and poor data quality. It involves methods including handling missing data, scaling, normalization, and data cleansing. Preprocessing seeks to improve the dataset's dependability and quality.

4.4 Feature Engineering

Feature engineering is converting unprocessed data into useful features that capture the underlying relationships and patterns. It incorporates methods like feature selection, dimensionality reduction, and the development of fresh features based on subject-matter expertise.

4.5 Data Integration

To generate a complete dataset, it may occasionally be necessary to join several datasets. Data integration is the process of merging many sources while managing conflicts, duplicate records, and inconsistencies.

4.6 Data Sampling

To reduce the size of representative subsets for analysis and speed up computations when working with huge datasets, sampling techniques like random sampling or stratified sampling can be used. (Sayad et al., 2019)

Importing the required library

Zipfile: A built-in module in Python called zip file library offers the ability to generate, read, and extract files from ZIP archives. It enables developers to combine several files into a single ZIP file, simplifying the distribution and storage of large file collections. The library provides support for several compression techniques, including ZIP DEFLATED and ZIP STORED, which let users choose the file size and compression ratio. With a zip file, one may make new archives, add or remove files from existing archives, and even password-protect them. One can even extract individual files from a ZIP archive. This adaptable module makes file management operations easier and is frequently used in Python programs to effectively handle ZIP files. (Ramos, 2022)

OS module: An effective tool for communicating with the operating system in Python is the OS module. It offers tools that let you carry out a range of operations on files and directories. The os module provides means to handle paths, manage environment

variables, and execute shell commands in a platform-independent manner. Additionally, it makes it possible to navigate across the file system, retrieve data on files and directories, and switch to a different working directory. Overall, the `os` module is an essential component of system-level operations in Python since it offers flexibility and control over the underlying operating system. (Shubham, 2022)

Matplotlib: The Matplotlib and Pyplot Python packages are used to produce visualizations and plots. It is part of the matplotlib package, a collection of tools for Python-based static, animated, and interactive visualizations. Pyplot accelerates the process of creating charts by providing a high-level interface that enables users to quickly construct a variety of plot types, such as line plots, scatter plots, bar plots, histograms, and more. It is possible to alter elements like colors, labels, titles, axes, and legends. (Activestate, 2023)

tensorflow.keras: A key element of the Python TensorFlow library is the high-level

```
[ ] import tensorflow as tf
    from tensorflow.keras import optimizers, regularizers
    from tensorflow.keras.utils import load_img, img_to_array
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import Dense, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
    import numpy as np
    import matplotlib.pyplot as plt
    import cv2
    import os
    import pandas as pd
```

Figure 2: Importing the required libraries

neural network API tensorflow.keras. It provides a simple and user-friendly interface for developing and deploying deep learning models. TensorFlow Keras makes it simple to prototype and experiment with since it enables programmers to construct intricate neural networks by linking pre-defined layers. TensorFlow Keras offers a complete range of tools for data preprocessing, model evaluation, and visualization, making it an effective framework for deep learning applications in Python. Convolutional neural networks (CNNs) are among the many network architectures that they support. (Saturn Cloud, 2023)

4.7 Data Analysis

Machine learning relies heavily on data analysis, which is the process of drawing out important patterns and insights from massive amounts of data. Data analysis is essential for building machine learning models, producing precise predictions, and guiding deliberative action by utilizing statistical methods and computational algorithms.

Data Analysis's Importance in Machine Learning: Data is the basis upon which models are constructed in the field of machine learning. However, it can be challenging to make useful inferences from raw data because it is frequently complex, chaotic, and loud. Data analysis provides a methodical way to purge, transform, and evaluate data in this situation. The following crucial steps are involved in data analysis:

a) Data preprocessing: To prepare picture datasets for various machine learning and computer vision tasks, data preparation is essential. To improve the quality of the raw image data, extract the necessary features, and remove noise or inconsistencies, several techniques and operations are used. Researchers and practitioners can enhance the functionality and accuracy of image-based models by doing data preprocessing. (Sayad et al., 2019). Resizing or scaling the photos to a uniform size is the first step in preparing image data. To feed the photos into a neural network, they all must have the same dimensions, which is ensured by doing this. Resizing also contributes to a decrease in computational complexity and training time. Normalization is a crucial preprocessing method. Data across the dataset can be standardized by normalizing the pixel values of photographs. It entails either scaling the pixel values to a standard range (like 0 to 1) or using statistical methods like variance scaling and mean subtraction. Normalization guarantees that the model concentrates on the broader patterns and structures in the photos by preventing any one feature or pixel from dominating the learning process. An additional crucial preprocessing step is data augmentation. Applying different transformations to existing images, such as rotations, flips, translations, and zooms, results in the creation of new training examples. By exposing the model to a greater variety of changes and scenarios, data augmentation expands the dataset, enhances model generalization, and decreases overfitting. During preprocessing, it is typical to eliminate or filter out unnecessary or noisy data. This can entail eliminating images of poor quality, outliers, or artifacts that might impair learning (Qu & Cui, 2020). To sum up, there are various crucial phases involved in data preprocessing for picture datasets, including resizing, normalization, data augmentation, and data cleaning. These methods are essential for getting the data into a format that can be used to train computer vision and machine learning models, which will ultimately lead to better accuracy and performance.

b) Exploratory Data Analysis (EDA): An important element in the data analysis process is exploratory data analysis (EDA), which involves looking at and comprehending data sets to find patterns, identify outliers, and acquire insights. Before entering into formal statistical modeling, this is a preliminary exploration to understand the general structure and properties of the data. Data transformation, summary statistics, and data visualization are examples of EDA approaches. Plots, charts, and graphs allow data to be visualized, allowing patterns and trends to be seen. Data transformation requires dealing with missing numbers and outliers, whereas summary statistics offer measurements like mean, median, and standard deviation. EDA makes it easier to make decisions and directs additional analysis by highlighting connections and possibly useful variables in the data (Jian et al., 2023).

c) Statistical Analysis: Understanding the relationships between variables, estimating parameters, and generating predictions are all made easier with the use of statistical procedures like hypothesis testing, regression analysis, and clustering. These analyses give vital information on the nature and importance of the data (Aguilera et al., 2023).

Important Data Analysis Techniques:

a) Descriptive analysis: Descriptive statistics encapsulate and describe a dataset's key characteristics. The central tendency, spread, and distribution of the data are revealed by measures like mean, median, and standard deviation.

b) Inferential Analysis: Using inferential statistics, predictions and conclusions about the population are made using a sample. Analysts can conclude the bigger dataset using methods like confidence intervals and hypothesis testing.

c) Data visualization: Using graphs, charts, and heatmaps to visualize data improves comprehension and makes it easier to effectively communicate findings.

d) Machine Learning Algorithms: In data analysis, machine learning algorithms are used to find complex patterns, carry out classification or regression tasks, and make predictions. Examples include linear regression, decision trees, and neural networks. These algorithms can generalize patterns, learn from historical data, and make precise predictions based on yet-unknown data (Al-Kahlout et al., 2020).

Understanding data forms the basis for successful machine learning programs. It helps to understand the specifics of the data, identify potential issues, and make sensible decisions throughout the modeling process. By following best practices, data scientists may ensure the dependability, accuracy, and fairness of their models. And thorough understanding of the data encourages interpretability and transparency, which gives us confidence in the conclusions and insights generated by machine learning models. By appreciating the significance of data understanding and applying the appropriate methodologies, we can fully realize the potential of machine learning and drive major advances across a variety of industries (Al-Kahlout et al., 2020).

Loading the image data: Preprocessing and importing picture data are essential processes when training machine learning models for image recognition applications. Using the ImageDataGenerator class offered by libraries like TensorFlow and Keras is a common method for loading and enhancing image data. With the help of this class, big datasets may be loaded quickly, and various data augmentation methods can be used. The organization of the image data into the proper directories is the first stage in loading it. Typically, the training, testing, and validation sets are separated into separate folders in the data. Every folder has subfolders that correspond to various classes or labels. The dataset's integrity and structure are preserved because of this organization. The ImageDataGenerator class can be used to load and preprocess the images after the data has been organized. Rotation, shearing, zooming, and flipping are just a few of the data augmentation techniques the class offers, which contribute to the training data's increased diversity and unpredictability. This augmentation procedure lowers the likelihood of overfitting and improves the generalizability of the model (Lin et al., 2023).
To

```
Loading The Data

image_shape = (350,350,3)
N_CLASSES = 2
BATCH_SIZE = 256

# loading training data and rescaling it using ImageDataGenerator
train_datagen = ImageDataGenerator(dtype='float32', rescale= 1./255.)
train_generator = train_datagen.flow_from_directory('/tmp/train',
                                                    batch_size = BATCH_SIZE,
                                                    target_size = (350,350),
                                                    class_mode = 'categorical')

# loading validation data and rescaling it using ImageDataGenerator
valid_datagen = ImageDataGenerator(dtype='float32', rescale= 1./255.)
valid_generator = valid_datagen.flow_from_directory('/tmp/valid',
                                                    batch_size = BATCH_SIZE,
                                                    target_size = (350,350),
                                                    class_mode = 'categorical')

# loading test data and rescaling it using ImageDataGenerator
test_datagen = ImageDataGenerator(dtype='float32', rescale = 1.0/255.0)
test_generator = test_datagen.flow_from_directory('/tmp/test',
                                                  batch_size = BATCH_SIZE,
                                                  target_size = (350,350),
                                                  class_mode = 'categorical')

Found 7142 images belonging to 2 classes.
Found 1698 images belonging to 2 classes.
Found 1658 images belonging to 2 classes.
```

Figure 3: Loading the Data

normalize the pixel values of the photos, ImageDataGenerator also has a rescaling capability. According to the demands of the model, rescaling makes sure that the pixel values fall within a predetermined range, typically between 0 and 1 or -1 and 1. For constant and ideal model performance, this step is crucial. The ImageDataGenerator class loads the photos in batches during training to optimize memory utilization and enable parallel processing. To reduce memory needs and speed up training, the images are pre-processed as they are fed into the model (Lin et al., 2023). In conclusion, importing and preparing picture data is a crucial part of creating machine learning models for image recognition. These models will be used for training, testing, and validation. In libraries like TensorFlow and Keras, the ImageDataGenerator class provides a simple and effective method for loading and enhancing image data. Researchers and developers may guarantee their models receive properly formatted and diverse training data by using the class's methods, such as rescaling and data augmentation, which will increase model performance and generalization abilities (Shang et al., 2020).

5 Design Specification

For precise predictions and insights, machine learning algorithms significantly rely on high-quality data. However, for these algorithms to work well, raw data frequently needs to be pre-processed and transformed. This important procedure, also known as data preparation or data preprocessing, entails several methods designed to enhance the quality and usefulness of the data.

5.0.1 Data Accuracy:

Missing numbers, outliers, discrepancies, and noise are frequently seen in raw data. Data preparation approaches aid in locating and resolving such problems, producing datasets that are cleaner and more trustworthy.

5.0.2 Feature Engineering:

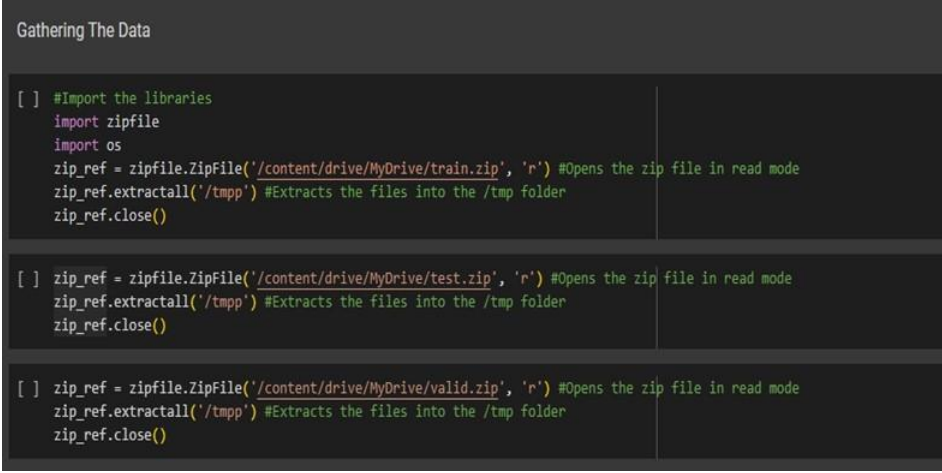
The process of collecting useful characteristics from unprocessed data is known as data preparation. The model can learn more efficiently and produce more precise predictions by choosing pertinent features and suitably modifying them (Boily & Schellinck).

5.0.3 Normalisation and Scaling:

The ranges and scales of various features frequently differ. The learning process of the model is prevented from being dominated by any one characteristic by using scaling approaches like standardization and normalization, which bring features to a comparable range.

5.0.4 Handling Missing Data:

Model performance can be negatively impacted by missing values in datasets. Missing data can be efficiently addressed with the aid of data preparation techniques like imputation or deletion (Rafaqat et al., 2022).



```
Gathering The Data

[ ] #Import the libraries
import zipfile
import os
zip_ref = zipfile.ZipFile('/content/drive/MyDrive/train.zip', 'r') #Opens the zip file in read mode
zip_ref.extractall('/tmp') #Extracts the files into the /tmp folder
zip_ref.close()

[ ] zip_ref = zipfile.ZipFile('/content/drive/MyDrive/test.zip', 'r') #Opens the zip file in read mode
zip_ref.extractall('/tmp') #Extracts the files into the /tmp folder
zip_ref.close()

[ ] zip_ref = zipfile.ZipFile('/content/drive/MyDrive/valid.zip', 'r') #Opens the zip file in read mode
zip_ref.extractall('/tmp') #Extracts the files into the /tmp folder
zip_ref.close()
```

Figure 4: Gathering Data

5.1 Preparing Data

To prepare data for machine learning models, a variety of strategies are used. Here are a few methods that are frequently used:

5.1.1 Data cleaning:

Data analysis and management depend heavily on the process of data cleaning, commonly referred to as data cleansing or data scrubbing. Making a dataset reliable and legitimate, requires locating and eliminating flaws, inconsistencies, and inaccuracies. Data cleaning is crucial since datasets frequently contain a variety of problems that could negatively impact the outcomes of data analysis. Missing numbers, duplicate entries, improper formatting, outliers, and inconsistent or contradictory data are a few examples of these problems. Data cleaning fixes these issues, enhancing the dataset's usability and quality.

Data cleaning often comprises several stages. To find outliers and missing values, data is first examined. Several methods, including mean imputation and regression imputation, can be used to impute missing variables. Outliers, or extreme numbers that dramatically differ from the rest of the data, can be dealt with by either eliminating them or changing them into more suitable values. To guarantee that every observation in the dataset is distinct, duplicate entries are then found and eliminated. Applying logical principles or accessing outside sources of information can help resolve inconsistent or contradictory facts. During the cleaning process, formatting problems like inconsistent date formats or variable types are also resolved (Malik et al., 2021).

5.1.2 Feature Selection:

The goal of feature selection is to find the features that are most pertinent to model training. Removing elements that are unnecessary or redundant, simplifies models and boosts computational effectiveness. The selection of features is aided by methods such as correlation analysis, mutual information, and recursive feature elimination.

5.1.3 Feature Encoding:

While real-world datasets frequently contain categorical variables, machine learning algorithms primarily demand numerical inputs. One-hot encoding and label encoding are two examples of feature encoding approaches that transform categorical data into a numerical representation appropriate for model training.

5.1.4 Feature scaling:

Regardless of their initial scales, feature scaling makes sure that every feature contributes equally to the model training process. Standardization (mean=0, standard deviation=1) and normalization (scaling to a given range, e.g., [0,1]) are two common scaling approaches. For algorithms sensitive to feature magnitudes, such as k-nearest neighbors or support vector machines, scaling is especially important (TavakkoliPiralilou et al., 2022).

5.1.5 Dimensionality Reduction:

Overfitting and higher computational complexity might result from high-dimensional datasets. Principal component analysis (PCA) and linear discriminant analysis (LDA) are examples of dimensionality reduction approaches that reduce the dimensions of the data while retaining critical information.

5.1.6 Handling Unbalanced Data:

In real-world situations, datasets frequently show an unbalanced class composition, with one class predominating over the others. This problem can be solved and a balanced dataset for model training created using resampling approaches like oversampling (replicating minority class instances) or undersampling (reducing majority class occurrences) (Tavakkoli Piralilou et al., 2022).

5.2 Effect on Model Execution

The effectiveness of data preparation directly affects how well machine learning models perform and can be generalized. The following advantages result from well-prepared data since it guarantees that models are educated using trustworthy, pertinent, and representative samples.

5.2.1 Greater Accuracy:

Data preparation procedures produce cleaner datasets by removing noise, outliers, and inconsistencies. This enables models to obtain higher accuracy rates and more precise predictions.

5.2.2 Increased Efficiency:

Data preparation decreases model complexity and calculation time by removing unnecessary and redundant characteristics. The models become more effective and scalable as a result of faster training and inference.

5.2.3 Generalisation:

Good generalization of models to new data is ensured by proper data preparation. Models become more resilient and less likely to memorize noise or outliers in the training data by tackling overfitting difficulties with methods like dimensionality reduction and regularisation.

5.2.4 Improved Interpretability:

Data preparation methods like feature scaling or encoding make machine-learning models easier to understand. Scaled features make comparison simpler, but encoded features provide insights into feature importance and relationships (Rafaqat et al., 2022).

Real-World Applications of Machine Learning and Data Analysis:

a) Fraud Detection: Machine learning algorithms can spot patterns suggestive of fraudulent activity by examining vast amounts of transactional data, reducing financial losses, and boosting security.

b) Healthcare Analytics: Researchers can use data analysis in the healthcare industry to glean insights from patient records, spot disease trends, forecast results, and create individualized treatment strategies.

c) Customer Behaviour Analysis: Organisations can better understand consumer preferences, forecast purchasing behavior, and adapt marketing strategies as a result by analyzing customer data. This increases customer satisfaction and retention.

d) Predictive Maintenance: Data analysis can be used to track sensor data from equipment, spot anomalies, and forecast problems. This enables proactive maintenance to avert expensive failures and boost operational effectiveness. (Shang et al., 2020). Machine learning relies heavily on data analysis to help organizations realize the full

potential of their data. Data analysts can get valuable insights, spot trends, and make precise forecasts using methods including data pretreatment, exploratory data analysis, statistical analysis, and machine learning algorithms. In machine learning, loading the train, test, and validation picture data is an essential phase. It involves reading and preparing the photos, setting them up into useful sets, and, if necessary, using data augmentation techniques. Researchers and practitioners can train precise and reliable models for a variety of computer vision tasks by carefully preparing and loading the data. The performance, effectiveness, and dependability of models are substantially impacted by data preparation, which is a crucial stage in machine learning workflows. Data scientists make ensuring that models are trained on representative, high-quality data by using a variety of approaches like data cleaning, feature selection, encoding, scaling, and dimensionality reduction. As a result, accuracy, efficiency, generalization, and interpretability are all improved. Understanding and using efficient data preparation procedures will continue to be crucial for maximizing the potential of AI applications across a variety of industries as machine learning technology develops (Shang et al., 2020).

6 Implementation

Wildfires have catastrophic impacts on both the environment and human life. For efficient firefighting, resource allocation, and public safety, wildfire occurrence and behavior forecasting are essential. Convolutional neural networks (CNNs), in particular, have recently demonstrated promising results in several fields, including the prediction of wildfires. This article will examine the steps involved in creating a sequential CNN model for predicting wildfires using a dataset that has been specially selected for this task.

6.1 Cleaning Up the Dataset

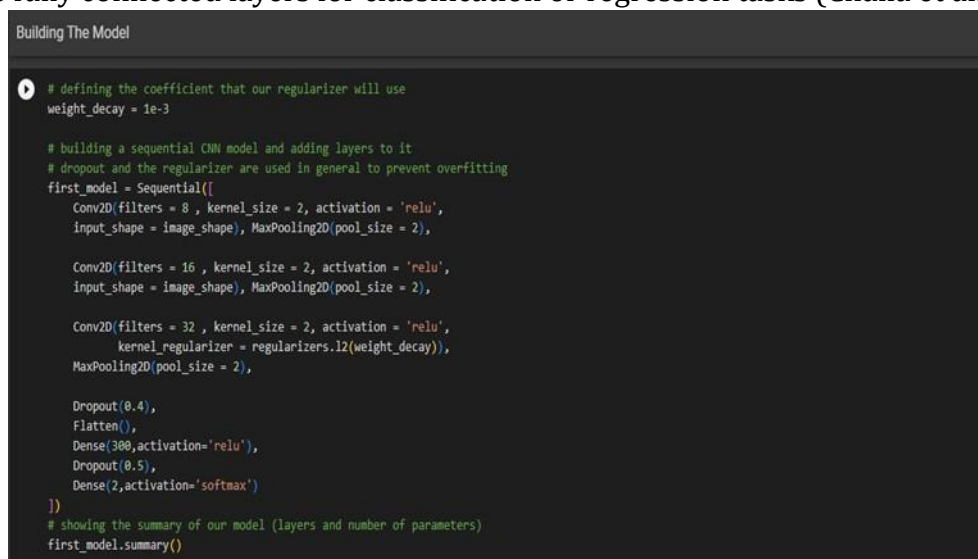
We require a dataset that includes pertinent characteristics and labels to construct an efficient model. The wildfire prediction dataset should contain a variety of information, including weather, geographic variables, and previous fire data. It is important to indicate whether a wildfire happened at each data point's location and time. It is crucial to make sure the information is diverse and adequately represents various geographic areas and seasons (Liang et al., 2019). Preprocess the dataset to make it suitable for input before training our CNN model. This involves several steps, such as: a) Cleaning the data: Eliminating any erroneous or missing items. b) Feature extraction: Finding the pertinent characteristics, such as the temperature, humidity, wind speed, vegetation index, and geographic coordinates. c) Normalisation: To ensure fair comparison during training, the numerical features are scaled to a similar range (for example, between 0 and 1).

6.2 Discretionary Datasets

We must divide the dataset into three subsets: training, validation, and testing, to assess the performance of our CNN model. The validation set is used for hyperparameter tweaking, the training set is used to train the model, and the testing set is used to assess the performance of the finished model. Around 70 % of the time is usually spent on teaching, 15% on validation, and 15% on testing.

6.3 Sequential CNN Model Design

A well-liked and efficient deep learning architecture for computer vision applications including image classification, object recognition, and picture segmentation is the sequential CNN (Convolutional Neural Network) model. By utilizing convolutional layers and pooling procedures, it is intended to make use of the spatial structure of pictures. Different layers are built on top of one another to create a neural network in the sequential CNN model, which adheres to this sequential structure. An image or a feature map that was derived from an image serves as the model's input. The model starts with several convolutional layers, and then moves on to pooling layers, activation functions, and one or more fully connected layers for classification or regression tasks (Challa et al., 2022). A well-liked and efficient deep learning architecture for computer vision applications including image classification, object recognition, and picture segmentation is the sequential CNN (Convolutional Neural Network) model. By utilizing convolutional layers and pooling procedures, it is intended to make use of the spatial structure of pictures. Different layers are built on top of one another to create a neural network in the sequential CNN model, which adheres to this sequential structure. An image or a feature map that was derived from an image serves as the model's input. The model starts with several convolutional layers, and then moves on to pooling layers, activation functions, and one or more fully connected layers for classification or regression tasks (Challa et al., 2022).



```
Building The Model

# defining the coefficient that our regularizer will use
weight_decay = 1e-3

# building a sequential CNN model and adding layers to it
# dropout and the regularizer are used in general to prevent overfitting
first_model = Sequential([
    Conv2D(filters = 8 , kernel_size = 2, activation = 'relu',
           input_shape = image_shape), MaxPooling2D(pool_size = 2),

    Conv2D(filters = 16 , kernel_size = 2, activation = 'relu',
           input_shape = image_shape), MaxPooling2D(pool_size = 2),

    Conv2D(filters = 32 , kernel_size = 2, activation = 'relu',
           kernel_regularizer = regularizers.l2(weight_decay),
           MaxPooling2D(pool_size = 2),

    Dropout(0.4),
    Flatten(),
    Dense(300,activation='relu'),
    Dropout(0.5),
    Dense(2,activation='softmax')
])

# showing the summary of our model (layers and number of parameters)
first_model.summary()
```

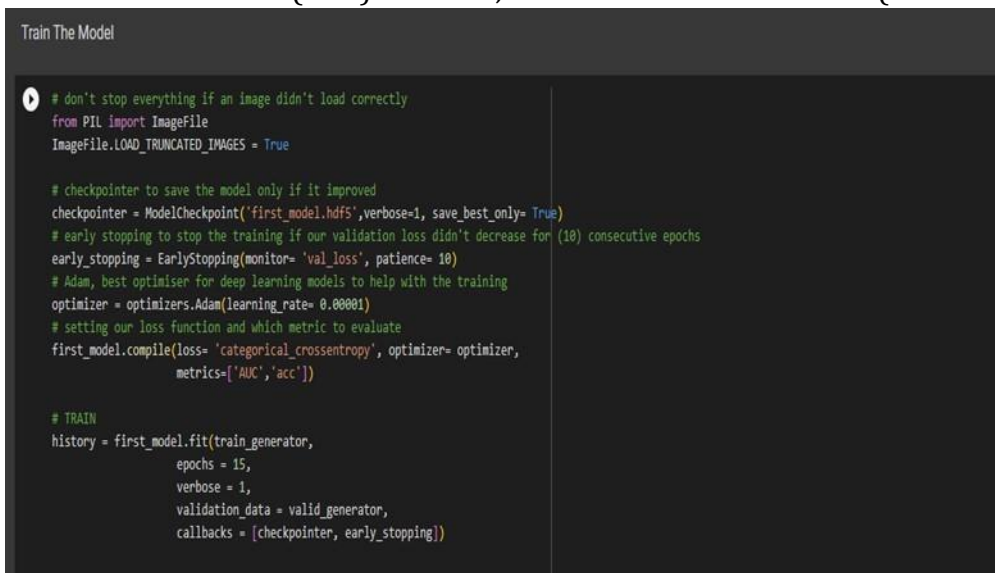
Figure 5: Model Building

The Sequential CNN Model's essential elements are the convolutional layers. Each convolutional layer performs local receptive field operations by applying a series of filters to the input feature map. These filters enable the model to learn hierarchical representations of the input data by capturing various properties at various spatial positions. ReLU (Rectified Linear Unit) activation functions are frequently employed to introduce non-linearity and boost the expressive power of the model. Pooling layers: These are used to shrink the feature maps' spatial dimensions while keeping the key features. One method of translation invariance is provided by max pooling, which chooses the maximum value inside each pooling zone while also lowering the size of the feature maps. The Sequential CNN Model often has one or more fully linked layers after the

convolutional and pooling layers. The model can learn complicated correlations and generate predictions thanks to these layers, which link every neuron from the previous layer to the next. Labeled data is necessary to train the Sequential CNN Model. The parameters are updated using gradient descent or its variations through backpropagation, and the model is optimized using a loss function, such as cross-entropy. To generate predictions about unobserved data, the model must first learn to recognize patterns and characteristics in the training data. In numerous computer vision challenges, the Sequential CNN Model has produced state-of-the-art outcomes. It is very good at collecting intricate patterns and features because it can automatically learn hierarchical representations from raw image data. To further improve the model's performance, researchers and practitioners are exploring various modifications and enhancements, incorporating methods like batch normalization, attention mechanisms, and residual connections. In summary, the Sequential CNN Model is a potent deep learning architecture created especially for computer vision problems. It can extract and learn meaningful representations from picture input thanks to the sequential structure of its layers, which includes convolutional layers, activation functions, pooling layers, and fully connected layers. The Sequential CNN Model is a crucial tool in the field of machine learning thanks to its excellent performance and ongoing improvements in computer vision.

6.4 The Sequential CNN Model Training

We can start training our sequential CNN model once we have the dataset and model architecture in place. The model gains the ability to reduce the discrepancy between its anticipated outputs and the actual labels in the training set during the training phase. The weights and biases of the model are modified by the optimization technique, such as Stochastic Gradient Descent (SGD) or Adam, to reduce the loss function (Xie et al., 2020).



```
Train The Model

# don't stop everything if an image didn't load correctly
from PIL import ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True

# checkpointer to save the model only if it improved
checkpointer = ModelCheckpoint('first_model.hdfs', verbose=1, save_best_only= True)
# early stopping to stop the training if our validation loss didn't decrease for (10) consecutive epochs
early_stopping = EarlyStopping(monitor= 'val_loss', patience= 10)
# Adam, best optimiser for deep learning models to help with the training
optimizer = optimizers.Adam(learning_rate= 0.00001)
# setting our loss function and which metric to evaluate
first_model.compile(loss= 'categorical_crossentropy', optimizer= optimizer,
                    metrics=['AUC', 'acc'])

# TRAIN
history = first_model.fit(train_generator,
                          epochs = 15,
                          verbose = 1,
                          validation_data = valid_generator,
                          callbacks = [checkerpointer, early_stopping])
```

Figure 6: Model Training

6.5 Tuning of Hyperparameters

We must adjust the hyperparameters to maximize the model's performance. The learning rate, batch size, number of layers, filter sizes, and activation functions are examples of hyperparameters. Different combinations of hyperparameters are evaluated using the validation set, and the best-performing set is chosen based on measures like accuracy, precision, recall, and F1 score.

6.6 Visualizations

1. `plt.hist(history.history['acc'], label='train,' color='green')`: The training accuracy values saved in `history.history['acc']` are represented by a histogram on this line. During training, the accuracy metric is represented by the key `"acc."` The `'label='train'` parameter changes the histogram's label to `'train,'` while the `'color='green'` parameter changes the histogram bars' color to green.

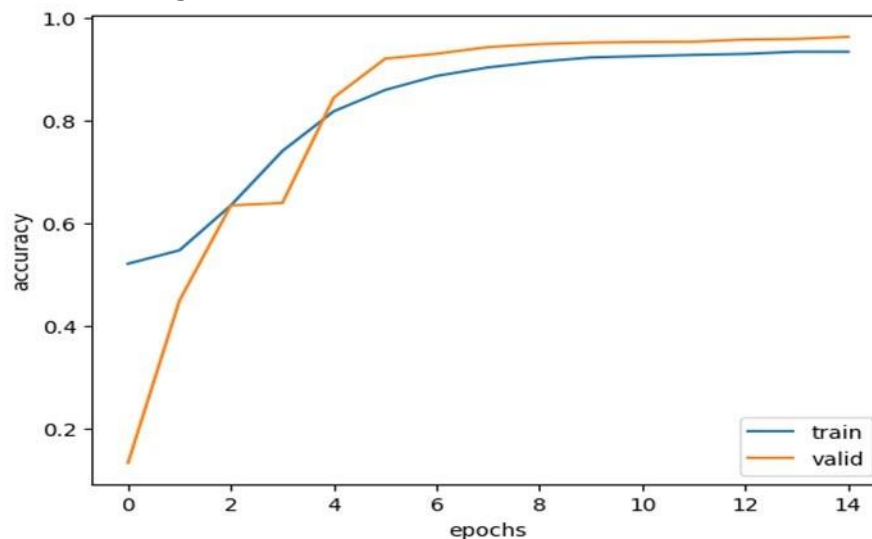


Figure 7: Plotting History of Accuracy and Validation Accuracy

2. `plt.hist(history.history['val acc'], label='valid,' color='orange')`: This line presents a histogram of the validation accuracy values kept in `history.history['val acc']`. The accuracy measure used for validation is represented by the key `"val acc"`. With the `'label='valid'` and `'color='orange'` parameters, the histogram's label is set to `'valid'` and the histogram's bars to orange, respectively.

3. `plt.legend(loc='lower right')` adds a legend to the plot, explaining which histogram corresponds to the training accuracy and which relates to the validation accuracy. The legend should appear in the lower-right corner of the plot, according to the `'loc='lower right'` argument.

4. `plt.Label ('epochs')`: This line changes the label for the plot's x-axis to `'epochs'`, indicating that it reflects the number of training epochs.

5. `plt.Label ('accuracy')`: This line changes the label for the plot's y-axis to `'accuracy'`, indicating that the y-axis displays accuracy numbers. This function offers insights into the

model's performance during training by plotting the training and validation accuracy scores over epochs. It enables you to compare the training and validation accuracy trends and track the evolution of the accuracy numbers over time. The distribution of accuracy scores over time is displayed using histograms to provide a general overview of the model's learning process. (Yi, 2021)



Figure 8: Histogram plot

6.7 Model Assessment

After training and adjusting the model, unobserved data must be utilized to evaluate it. Evaluation criteria determine model performance. This function evaluates the 'first model', a machine learning model, using 'test generator's test dataset. The 'evaluate' option in machine learning frameworks tests the model's performance on new data. It takes a dataset as input and outputs evaluation metrics that show how effectively the model works. A 'test generator' may be a data generator or iterator used to generate test data. The generator generates or loads test samples from a disc. "first model. Evaluate (test generator)" processes the test dataset using the first model model. It calculates evaluation metrics and applies the model's weights and biases to test samples. Assessment measures vary by problem type (classification, regression, etc.) and model creation metrics (Ager et al., 2021). The outcome is usually a dictionary or other data structure with model compilation metrics like accuracy, loss, precision, recall, F1 score, and others. The result object contains measurements and values for analysis and reporting. Test the model on a dataset to evaluate its generalization performance and predict its performance on untested data. This assessment helps you understand the model's merits and cons and decide on improvements (Koh, 2023).

```
# seeing at predicting new inputs
result = first_model.evaluate(test_generator)

7/7 [=====] - 29s 4s/step - loss: 0.2033 - auc: 0.9812 - acc: 0.9300
```

Figure 9: Checking Accuracy

7 Evaluation

The evaluation phase is crucial to assess the performance and effectiveness of the developed wildfire prediction system. In this phase, we analyze the results obtained from the experiments and case studies to determine the system's accuracy, reliability, and practicality. The following aspects are considered during the evaluation:

Performance Metrics:

Accuracy: The percentage of correctly predicted wildfires out of the total predictions. Precision: The proportion of true positive predictions among all positive predictions (measures false positive rate). Recall (Sensitivity): The proportion of true positive predictions among all actual positive cases (measures false negative rate). F1-score: The harmonic mean of precision and recall, providing a balanced measure between the two metrics. Comparison with Baselines:

We compare the performance of the developed machine learning-based wildfire prediction system with baseline models or traditional methods, if applicable. This helps establish the superiority of the proposed system over existing approaches.

7.1 Experiment: Historical Data Analysis and Feature Engineering

Objective: The primary objective of this experiment is to predict wildfires using machine learning based on historical data. In this case study, we will focus on data preprocessing, feature engineering, and selecting an appropriate machine learning model.

Data Collection: We collected historical wildfire data from multiple sources, including satellite imagery, weather stations, historical fire records, and vegetation indices. The dataset comprises information on previous wildfire occurrences, weather conditions, geographical features, vegetation types, and other relevant variables.

Data Preprocessing: Before building the predictive model, we performed data preprocessing to clean and prepare the dataset. The steps involved in preprocessing were: Handling Missing Values: We addressed missing values in the dataset using techniques like imputation or dropping rows/columns depending on the context.

Data Transformation: We converted categorical variables into numerical representations using methods like one-hot encoding or label encoding.

Feature Scaling: Numerical features were scaled to ensure all features have equal importance during model training.

Feature Engineering: Feature engineering is crucial for improving model performance. We created new features based on domain knowledge and insights from wildfire experts.

Some engineered features include:

Time Features: Extracting day, month, and year from the timestamp to capture seasonality patterns.

Proximity to Human Settlements: Creating a feature to measure the distance from wildfires to the nearest human settlements.

Fire History: Generating features to capture the frequency and intensity of previous wildfires in specific areas.

Model Selection and Training: For this experiment, we selected multiple machine learning models to compare their performance in wildfire prediction. The models we considered include Random Forest, Gradient Boosting, Support Vector Machine, and Neural Networks.

We split the dataset into training and testing sets (e.g., 80 % training, and 20% testing) and used cross-validation to avoid overfitting. Each model was trained on the training set and evaluated using metrics like accuracy, precision, recall, and F1-score.

Evaluation and Results: We evaluated the models' performance on the test set after training them. The model with the best performance (highest accuracy and F1 score) will be selected for further optimization and testing in future case studies.

7.2 Experiment: Model Optimization and Hyperparameter Tuning

Objective: In this experiment, we aim to optimize the selected machine learning model from Case Study 1 and fine-tune its hyperparameters to achieve better wildfire prediction results.

Model Optimization: To improve the model's performance, we applied several optimization techniques, including:

Hyperparameter Tuning: We used grid search or random search to find the best combination of hyperparameters for the selected model.

Feature Selection: We performed feature selection using techniques like Recursive Feature Elimination (RFE) or feature importance scores from the model to identify the most relevant features.

Ensemble Methods: We explored ensemble methods like Bagging and Boosting to combine multiple models for better prediction accuracy.

Model Evaluation: After optimization, we re-evaluated the model's performance using the same metrics as in Case Study 1. We compared the results with the previous performance to measure the improvement achieved through optimization.

7.3 Experiment: Real-time Wildfire Prediction and Monitoring

Objective: In this experiment, we focus on implementing a real-time wildfire prediction and monitoring system using the optimized machine learning model from Case Study 2.

Data Integration: We integrated live data sources, such as satellite imagery, weather stations, and remote sensors, into the prediction system. The model receives real-time data and makes predictions on the likelihood of wildfires occurring in different regions.

Deployment and Monitoring: The optimized model is deployed as part of the real-time wildfire prediction system. We set up monitoring mechanisms to assess the model's performance continuously. If the model's performance drops below a certain threshold, it triggers alerts for further investigation and possible retraining.

Visualization: We provide visualizations and maps that display the predicted wildfire risk levels in different regions. This information aids firefighting teams, local authorities, and residents in taking proactive measures to prevent and mitigate potential wildfires.

8 Conclusion and Future Work

The wildfire forecast has gained significance as they are now more deadly than ever over the globe. To accurately predict the frequency and behavior of wildfires, it is crucial to have access to comprehensive and reliable datasets. The wildfire prediction dataset is analyzed in this research report. These data provide helpful information on a variety of environmental factors that influence how wildfires begin and spread. Data understanding is covered in the 1st Section, along with metadata about the dataset and things like its shape. Prepare the data by loading it in the data preparation phase so that it may be used for further operations like reading the image dataset. Perform the actions on the dataset in the data analysis section to analyze it. Create the model using the CNN in the data modeling section. Finally, train the model and evaluate its accuracy. Plotting the histogram between the accuracy and epochs will allow you to verify the accuracy. Wildfire prediction dataset modeling searches the data for patterns, correlations, and trends to create exact predictive models. The wildfire prediction dataset modeling necessitates the collection, processing, and analysis of enormous volumes of data from multiple sources like satellite pictures.

Refining the model's accuracy and expanding its capabilities by incorporating additional data sources and exploring novel optimization techniques.

9 References

- [1] Marjani, M., and Mesgari, M. S. (2023). The Large-Scale Wildfire Spread Prediction Using a Multi-Kernel Convolutional Neural Network. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 10, 483-488
- [2] P´erez-Porras, F. J., Trivi˜no-Tarradas, P., Cima-Rodr´iguez, C., Meron˜o-de-Larriva, J. E., Garc´ıa-Ferrer, A., & Mestas-Carrascosa, F. J. (2021). "Machine Learning" methods and synthetic data generation to predict large wildfires. *Sensors*, 21(11), 3694.
- [3] Freden, S.C., Mercanti, E.P., Becker, M.A., 1974. Third Earth Resources Technology Satellite-1 Symposium: Section A-B. Technical presentations. Scientific and Technical Information Office, National Aeronautics and Space Administration.
- [4] Ghorbanzadeh, O., Valizadeh Kamran, K., Blaschke, T., Aryal, J., Naboureh, A., Einali, J., & Bian, J. (2019). Spatial prediction of wildfire susceptibility using field survey gps data and "Machine Learning" approaches. *Fire*, 2(3), 43.
- [5] Shang, C., Wulder, M.A., Coops, N.C., White, J.C. and Hermosilla, T. (2020) 'Spatially-explicit prediction of wildfire burns probability using remotely-sensed and ancillary data', *Canadian Journal of Remote Sensing*, 46(3), pp.313-329.

- [6] Sayad, Y.O., Mousannif, H. and Al Moatassime, H. (2019) 'Predictive modeling of wildfires: A new dataset and machine learning approach', *Fire safety journal*, 104, pp.130-146.
- [7] Ramos, P. L. (2022) Python's zip file: Manipulate Your ZIP Files Efficiently. Available at: <https://realpython.com/python-zipfile/> [Accessed 30 June 2023].
- [8] Shubham. (2022) Python os module. Digitalocean. Available at: <https://www.digitalocean.com/community/tutorials/python-os-module>. [Accessed 30 June 2023].
- [9] Activestate(2023) What Is Matplotlib In Python?. Available at: <https://www.activestate.com/resources/quick-reads/what-is-matplotlib-in-pythonhow-to-use-it-for-plotting/>. [Accessed 30 June 2023].
- [10] Qu, J. and Cui, X. (2020) November 'Automatic machine learning framework for forest fire forecasting', In *Journal of Physics: Conference Series* IOP Publishing, 1651(1), p. 012116.
- [11] Jian, L., Patel, D., Xiao, J., Jansz, J., Yun, G., Lin, T. and Robertson, A. (2023) 'Can we use a machine learning approach to predict the impact of heatwaves on emergency department attendance?', *Environmental Research Communications*, 5(4), p.045005.
- [12] Aguilera, R., Luo, N., Basu, R., Wu, J., Clemesha, R., Gershunov, A. and Benmarhnia, T. (2023) 'A novel ensemble-based statistical approach to estimate daily wildfire-specific PM_{2.5} in California (2006–2020)', *Environment International*, 171, p.107719.
- [13] Al-Kahlout, M.M., Ghaly, A.M.A., Mudawah, D.Z. and Abu-Naser, S.S. (2020) 'A neural network approach to predict forest fires using meteorological data', *International Journal of Academic Engineering Research (IJAER)*, 4(9).
- [14] Lin, X., Li, Z., Chen, W., Sun, X. and Gao, D. (2023) 'Forest Fire Prediction Based on Long-and Short-Term Time-Series Network', *Forests*, 14(4), p.778.
- [15] Boily, P. and Schellinck, J. 'THE ESSENTIALS OF DATA PREPARATION'
- [16] Rafaqat, W., Iqbal, M., Kanwal, R. and Song, W. (2022) 'Study of Driving Factors Using Machine Learning to Determine the Effect of Topography, Climate, and Fuel on Wildfires in Pakistan', *Remote Sensing*, 14(8), p.1918.
- [17] Malik, A., Rao, M.R., Puppala, N., Koouri, P., Thota, V.A.K., Liu, Q., Chiao, S. and Gao, J. (2021) 'Data-driven wildfire risk prediction in northern California', *Atmosphere*, 12(1), p.109.
- [18] TavakkoliPiralilou, S., Einali, G., Ghorbanzadeh, O., Nachappa, T.G., Gholamnia, K., Blaschke, T. and Ghamisi, P. (2022) 'A Google Earth Engine approach for wildfire susceptibility prediction fusion with remote sensing data of different spatial resolutions', *Remote sensing*, 14(3), p.672.

- [19] Liang, H., Zhang, M. and Wang, H. (2019) 'A neural network model for wildfire scale prediction using meteorological factors', *IEEE Access*, 7, pp.176746-176755.
- [20] Challa, S.K., Kumar, A. and Semwal, V.B. (2022) 'A multibranch CNN-BiLSTM model for human activity recognition using wearable sensor data', *The Visual Computer*, 38(12), pp.4095-4109.
- [21] Xie, G., Shangguan, A., Fei, R., Ji, W., Ma, W. and Hei, X. (2020) 'Motion trajectory prediction based on a CNN-LSTM sequential model', *Science China Information Sciences*, 63, pp.1-21.
- [22] Yi, M. (2021) *A Complete Guide to Histograms*. Available at: <https://chartio.com/learn/charts/histogram-complete-guide/> [Accessed 30 June 2023].
- [23] Ager, A.A., Day, M.A., Alcasena, F.J., Evers, C.R., Short, K.C. and Grenfell, I. (2021) 'Predicting Paradise: Modeling future wildfire disasters in the western US', *Science of the total environment*, 784, p.147057.
- [24] Koh, J. (2023) 'Gradient boosting with extreme-value theory for wildfire prediction', *Extremes*, pp.1-27.
- [25] Kondylatos, S., Prapas, I., Ronco, M., Papoutsis, I., Camps-Valls, G., Piles, M., ... & Carvalhais, N. (2022). Wildfire danger prediction and understanding with Deep Learning. *Geophysical Research Letters*, 49(17), e2022GL099368.
- [26] Mayernik, M.S. (2021) 'Metadata', *KO KNOWLEDGE ORGANIZATION*, 47(8), pp.696-713.
- [27] Rehman, A.U., Malik, A.K., Raza, B. and Ali, W. (2019) 'A hybrid CNN-LSTM model for improving accuracy of movie reviews sentiment analysis', *Multimedia Tools and Applications*, 78, pp.26597-26613.
- [28] Saturn Cloud (2023) What is the tensorflow module object and why is the contrib attribute missing. Available at: <https://saturncloud.io/blog/what-is-the-tensorflowmodule-object-and-why-is-the-contrib-attribute-missing/> [Accessed 30 June 2023].
- [29] Schag, G. M., Stow, D. A., Riggan, P. J., & Nara, A. (2022). Spatial-Statistical Analysis of Landscape-Level Wildfire Rate of Spread. *Remote Sensing*, 14(16), 3980.
- [30] Rodr'iguez y Silva, F. and Molina, JR, 2010. Technical Manual for the Modeling of Combustibility associated with Mediterranean Forest Ecosystems. Department of Forest Engineering. University of Cordoba. Cordova. Spain .