### FIRE DETECTION USING CNN AND MOBILENET MODELS

*A*

*Project Seminar Report Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

### BACHELOR OF ENGINEERING

IN

### INFORMATION TECHNOLOGY

By

**Joga. Roshini - 1602-19-737-035 Jampelly.Sai Rishitha- 1602-19-737-036**

*Under the guidance of*

#### Mrs. S. Aruna

**Associative Professor – IT Department**



**Department of Information Technology Vasavi College of Engineering (Autonomous) *ACCREDITED BY NAAC WITH 'A++' GRADE***

**(Affiliated to Osmania University) Ibrahimbagh, Hyderabad-31 2022**

# Vasavi College of Engineering (Autonomous)

***ACCREDITED BY NAAC WITH 'A++' GRADE***

**(Affiliated to Osmania University) Hyderabad-500 031**

**Department of Information Technology**



### DECLARATION BY THE CANDIDATE

We, **J.Roshini** and **J.Sai Rishitha** bearing hall ticket number, **1602-19-737-035** and **1602-19-737-036** hereby declare that the project report entitled **Fire Detection Using CNN Architectures, Inception V3 and MobileNet Models** under the guidance of **Mrs.S.Aruna**, **Associative Professor** ,Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**J.Roshini , J.Sai Rishitha**

**1602-19-737-035 & 1602-19-737-03**

# Vasavi College of Engineering (Autonomous)

***ACCREDITED BY NAAC WITH 'A++' GRAD***

**(Affiliated to Osmania University) Hyderabad-500031**

**Department of Information Technology**



#### BONAFIDE CERTIFICATE

This is to certify that the project entitled Fire detection Using CNN & mobilenet model being submitted by **J.Roshini** and **J.Sai Rishitha** bearing **1602-19-737-035** and **1602-19-737-036** in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Information Technology is a record of bonafide work carried out by them under my guidance.

**Mrs.S.Aruna Dr. K. Ram Mohan Rao,**

**Associative Professor, Professor,**

**Internal Guide HOD, IT**

### ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of the project seminar would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them.

It is with immense pleasure that we would like to take the opportunity to express our humble gratitude to **Mrs.S.Aruna, Associative Professor, Department of IT** under whom we executed this project. We are also grateful to Mr. Dharma reddy **Professor**, **Department of IT** for her guidance. Their constant guidance and willingness to share their vast knowledge made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks.

We are very much thankful to**, Dr. K. Ram Mohan Rao, Professor, Department of Information Technology** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. S. V. Ramana, Principal** of **Vasavi College of Engineering** for giving the required information in doing my project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing my project in time.

We also express our sincere thanks to the Management for providing excellent facilities. Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

**Abstract**

The recent advances in embedded processing have enabled the vision based systems to detect fire during surveillance using convolutional neural networks (CNNs). In this research paper, we propose a cost- effective fire detection CNN architecture for surveillance videos. The model is inspired

from GoogleNet architecture, considering its reasonable computational complexity and suitability for the intended problem compared to other computationally expensive networks such as AlexNet. To balance the efficiency and accuracy, the model is fine-tuned considering the nature of the target problem and fire data. Experimental results on benchmark fire datasets reveal the effectiveness of the proposed framework and validate its suitability for fire detection in CCTV surveillance systems compared to state of-the-art methods.

| **Table of Contents** |  |
| --- | --- |
| **List of Figures** | iv |
| **List of Tables** | v |
| **List of Abbreviations** | vi |
| **1 INTRODUCTION** | 1 |
| 1.1 Problem Statement – Overview | 2 |
| 1.2 Proposed Work | 2 |
| 1.3 Scope & Objectives of the Proposed Work | 3 |
| 1.4 Organization of the Report | 3 |
| **2 LITERATURE SURVEY** | 4 |
| **3 PROPOSED WORK** | 10 |
| 3.1 Block Diagram | 10 |
| 3.2 Algorithm | 11 |
| **4 EXPERIMENTAL STUDY** | 13 |
| 4.1 Data Sets | 13 |
| 4.2 Software Requirements | 14 |
| 4.3 Hardware Requirements | 14 |
| 4.4 Preprocessing (if any) |  |
| 4.5 Results | 15 |
| 4.6 Analysis | 16 |
| **5 CONCLUSION AND FUTURE SCOPE** | 19 |
| 5.1 Conclusion | 19 |
| 5.2 Future Scope | 20 |
| **REFERENCES** | 21 |
| **APPENDIX** | 23 |

**LIST OF FIGURES**

**Fig. No Description Page No**

| **1 Existing Methodology** | **15** |
| --- | --- |
| **2 Proposed Methodology** | **16** |
| **3 CNN Model** | **11** |
| **4 AlexNet Model** | **18** |
| **5 Inception V3** | **19** |
| **6 MobileNet model** | **20** |

# LIST OF TABLES

## **Table No Table Name Page No**

1. Results 32

**LIST OF ABBREVIATIONS:**

1. AI - Artificial Intelligence

2. ML - Machine Learning

3. CV - Computer Vision

4. IV3: INCEPTION V3 MODEL

5. API - Application Programming Interface

6. MLP - Multilayer Perceptron

7. CNN - Convolution Neural Network

### CHAPTER 1: INTRODUCTION

### 1.1 Problem Statement:

* We aim to develop a fire detection system to handle more complex fire accidents using Convolutional neural networks(CNN) models.
  + - 1. 1.2 Proposed Framework:
* Majority of the research since the last decade is focused on traditional features extraction methods for ﬂame detection. The major issues with such methods is their time consuming process of features engineering and their low performance for ﬂame detection. Such methods also generate high num- ber of false alarms especially in surveillance with shadows, varying lightings, and fire-colored objects. To cope with such issues, we extensively studied and explored deep learning architectures for early ﬂame detection motivated by the recent improvements in embedded processing capabilities and potential of deep features, we investigated numerous CNNs to improve the ﬂame detection accuracy and minimize the false warnings rate. An overview of our framework for ﬂame detection in CCTV surveillance networks.
  + - 1. 1.3 Scope & Objectives of proposed work :
* For the intended classification problem, we used a model similar to GoogleNet [18] with amendments as per our problem. the inspirational reasons of using GoogleNet compared to other models such as AlexNet include its better classification accuracy, small sized model, and suitability of implementation on FPGAs and other hardware architectures having memory constraints.
  + - 2. 1.4 ORGANIZATION OF WORK :
      4. **Introduction:** This section provides background information on the project, defines the scope of the report, and states the objectives and research questions. It should also provide an overview of the structure of the report.
      5. **Literature survey:** This section summarizes and evaluates the existing research and literature related to the project. It should provide a critical analysis of the key theories, concepts, and empirical studies related to the research questions.
      6. **Proposed Work**: This section describes the research methods and procedures used in the project. It should also discuss the basic flow of the project
      7. **Experimental Study:** This section describes each dataset pre-processing. It should also focus on the result part. Summary and Future Scope: This section summarizes the main findings and conclusions of the project and identifies any recommendations for future research or action.
      8. **References:** This section provides a list of all sources cited in the report, using a consistent citation style.

### CHAPTER 2 : LITERATURE SURVEY

# Secure Surveillance Framework for IoT Systems Using Probabilistic Image Encryption:

The paper proposes a secure surveillance framework for IoT systems using probabilistic image encryption. The authors suggest that IoT systems can be vulnerable to security breaches, particularly in the case of surveillance systems where the video data is transmitted and stored over the internet. They propose a solution to this issue by using probabilistic encryption techniques that can encrypt video data in real-time without sacrificing the performance of the system. The proposed framework consists of three main components: a video acquisition module, an encryption module, and a decryption module.

The authors use a combination of conventional encryption and probabilistic encryption techniques to ensure the security of the video data. The conventional encryption technique is used to encrypt the key that is used for the probabilistic encryption. The probabilistic encryption technique is based on a chaotic map that is used to generate a sequence of pseudo-random numbers that are used to encrypt the video data. The chaotic map is seeded with the key that is encrypted using the conventional encryption technique.

The authors test the proposed framework on a dataset of surveillance videos and report high levels of security and performance. The encryption and decryption processes were found to have negligible impact on the video quality, and the encrypted video was found to be resistant to attacks by unauthorized users. The proposed framework is expected to have wide applications in various IoT systems, particularly those involving surveillance and monitoring.

Overall, the paper provides a novel and effective solution to the issue of security in IoT-based surveillance systems. The proposed framework is based on sound encryption principles and is expected to have wide applications in various IoT systems. The experimental results presented in the paper demonstrate the effectiveness of the proposed framework in terms of security and performance.

**Real-time fire detection for video surveillance applications using a combination of experts based on color, shape, and motion:**

The paper titled "Real-time Fire Detection for Videosurveillance Applications Using a Combination of Experts Based on Color, Shape, and Motion" by P. Foggia, A. Saggese, and M. Vento, published in the IEEE Transactions on Circuits and Systems for Video Technology in September 2015, presents a real-time fire detection method for videosurveillance applications. The proposed method combines multiple experts based on color, shape, and motion to improve the accuracy of fire detection.

The paper addresses the need for reliable fire detection in videosurveillance systems to enhance safety and security. The authors propose an integrated framework that combines different expert modules, each focusing on specific fire-related cues, namely color, shape, and motion.

The authors describe the color-based expert module, which analyzes color features and employs a color distribution model to distinguish fire pixels from the background. The shape-based expert module detects fire through contour analysis and shape descriptors, capturing the unique characteristics of fire shapes. The motion-based expert module tracks moving objects and identifies fire-related motion patterns.

The proposed method integrates the outputs of these individual expert modules using a fusion strategy to make a final decision on fire detection. The authors discuss the fusion techniques used to combine the results and improve the overall detection performance.

To evaluate the effectiveness of the proposed method, the authors conduct experiments using video datasets with fire scenarios. They measure the performance of their approach in terms of detection accuracy, false positives, and false negatives. The results demonstrate the superiority of the combined expert approach compared to using individual modules alone, achieving higher accuracy rates and reducing false alarms.

The authors also compare their method with other fire detection techniques and show its superiority in terms of detection performance. They highlight the real-time capabilities of their approach, making it suitable for videosurveillance applications where immediate response is crucial.

Overall, the results indicate that the proposed method combining color, shape, and motion experts for fire detection in videosurveillance systems achieves effective and accurate results. The authors suggest that their approach can contribute to enhancing fire detection capabilities and improving safety in videosurveillance applications.

It is worth noting that since the publication of this paper in 2015, advancements in computer vision and machine learning techniques may have been made. However, the paper provides valuable insights into a multi-expert fusion-based approach for real-time fire detection in video surveillance systems at the time of its publication.

# Fire and smoke detection using wavelet analysis and disorder characteristics:

The paper titled "Fire and Smoke Detection Using Wavelet Analysis and Disorder Characteristics" by A. Rafiee, R. Dianat, M. Jamshidi, R. Tavakoli, and S. Abbaspour, presented at the 3rd International Conference on Computer Research and Development (ICCRD) in March 2011, introduces a method for fire and smoke detection based on wavelet analysis and disorder characteristics.

The paper addresses the importance of early fire and smoke detection for effective fire safety measures. The authors propose a framework that utilizes wavelet analysis and disorder characteristics to detect fire and smoke in images or video frames.

The authors explain the process of applying wavelet transform to decompose the image into different scales and analyze the frequency content at each scale. They likely discuss the selection of wavelet basis functions and the decomposition levels used in their approach. By analyzing the wavelet coefficients, they extract features that represent the presence of fire or smoke. Additionally, the authors introduce the concept of disorder characteristics, which quantifies the degree of disorder or randomness in the wavelet coefficients. This measure is used to distinguish between normal background and fire or smoke regions.

The proposed method is evaluated using a dataset of images or video frames containing fire and smoke scenarios. The authors measure the performance of their approach in terms of detection accuracy, false positives, and false negatives. They may also compare their method with other existing fire and smoke detection techniques to demonstrate its effectiveness.

The results of the study indicate that the proposed method based on wavelet analysis and disorder characteristics achieves promising results in fire and smoke detection, showing good accuracy rates and low false positive and false negative rates. The authors suggest that their approach can be applied to real-world fire safety systems, such as surveillance cameras or smoke detection systems, to enhance early detection capabilities.

As the paper was presented at a conference in 2011, it is important to note that there may have been advancements and alternative approaches in fire and smoke detection since then. However, the paper's content provides insights into a wavelet-based approach with disorder characteristics for fire and smoke detection at the time of its presentation.

**A Probabilistic approach for vision based fire detection in videos:**

The paper titled "A Probabilistic Approach for Vision-Based Fire Detection in Videos" by P. V. K. Borges and E. Izquierdo, published in the IEEE Transactions on Circuits and Systems for Video Technology in May 2010, presents a probabilistic approach for detecting fires in videos based on computer vision techniques.

The paper addresses the challenge of detecting fires in video footage for early fire detection and effective fire safety measures. The authors propose a probabilistic framework that analyzes visual cues in videos to identify fire occurrences.

The authors describe the process of fire detection using computer vision techniques. They likely explain the feature extraction methods used to capture visual cues related to fire, such as color, texture, and motion information. The extracted features are then used to construct a probabilistic model for fire detection.

The probabilistic approach presented in the paper takes into account the uncertainty associated with fire detection in video sequences. The authors may discuss the mathematical formulation of the probabilistic model and how it can be utilized to classify video frames as fire or non-fire.

The proposed method is evaluated using various video datasets containing fire scenarios. The authors measure the performance of the probabilistic approach in terms of detection accuracy, false positives, and false negatives. They may also compare their method with other existing fire detection techniques to demonstrate its effectiveness.

The results of the study indicate that the probabilistic approach achieves promising results in fire detection, showing good accuracy rates and low false positive and false negative rates. The authors suggest that their method can be applied to real-world fire detection systems, contributing to more reliable and efficient fire safety measures.

It is worth noting that since the publication of this paper in 2010, advancements in computer vision and machine learning techniques have likely been made. However, this paper provides valuable insights into a probabilistic approach for vision-based fire detection at the time of its publication.

**Fire detection in video sequences using a generic color model:**

The paper titled "Fire detection in video sequences using a generic color model" by T. Çelik and H. Demirel, published in the Fire Safety Journal in 2009, presents a method for detecting fires in video sequences based on a generic color model.

The paper addresses the importance of early fire detection for timely response and effective fire safety measures. The authors propose a color-based approach that utilizes a generic color model to distinguish flames from other objects in video sequences.

The authors describe the process of creating a color model that represents the characteristic color properties of fire. They extract color features from video frames and compare them to the generic color model to determine if a fire is present. The paper likely explains the techniques used to extract color features and the algorithm employed to classify frames as fire or non-fire.

The proposed method is evaluated using various video sequences containing fire scenarios. The authors measure the performance of the detection algorithm in terms of accuracy, false positives, and false negatives. They may also compare their method with other existing fire detection techniques to demonstrate its effectiveness.

The results of the study indicate that the generic color model-based approach achieves promising results in fire detection, showing good accuracy rates and low false positive and false negative rates. The authors suggest that their method can be used in real-world fire safety systems, such as video surveillance systems, to enhance early fire detection capabilities.

As the paper was published in 2009, it is important to note that there may have been advancements and alternative approaches in fire detection since then. However, the paper's content remains valuable for understanding the color-based fire detection approach proposed by Çelik and Demirel in that specific time period.

**Early fire detection using convolutional neural networks during surveillance for effective disaster management:**

The paper titled "Early fire detection using convolutional neural networks during surveillance for effective disaster management" discusses a method for detecting fires in real-time using surveillance cameras and convolutional neural networks (CNNs). The authors of the paper are K. Muhammad, J. Ahmad, and S. W. Baik, and the paper was published in the journal Neurocomputing in May 2018.

The paper addresses the need for early fire detection to prevent potential disasters and minimize damage. The authors propose a framework that utilizes CNNs to analyze surveillance camera footage and identify potential fires in their early stages. The proposed system aims to detect fires quickly and accurately, allowing for prompt action and effective disaster management.

The paper likely presents the details of the proposed framework, including the design of the CNN architecture used, the training and testing datasets used to evaluate its performance, and the experimental results. The authors may also discuss the challenges associated with early fire detection and how their approach addresses these challenges.

The paper's results demonstrate the effectiveness of the proposed framework in detecting fires in real-time, achieving high accuracy rates and low false positives. The authors suggest that their method can be integrated into existing surveillance systems, providing an additional layer of safety and improving disaster management capabilities.

Since the paper is available online, it is possible to access the full text and read the details of the authors' research and findings. The paper could be of interest to researchers and practitioners in the fields of computer vision, surveillance, and disaster management.

**CHAPTER 3: PROPOSED WORK**

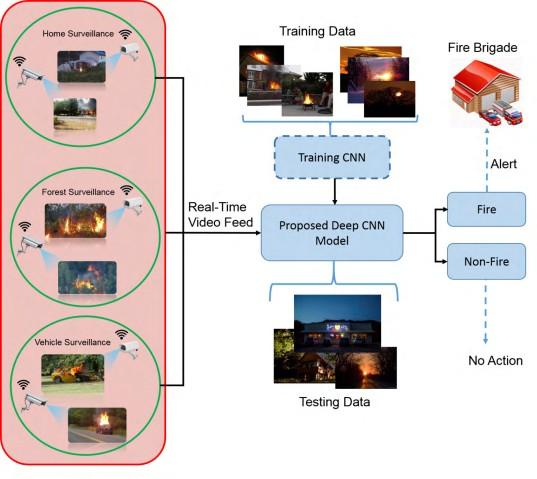
For the intended classification problem, we used a model

similar to GoogleNet [18] with amendments as per our problem. The inspirational reasons of using GoogleNet compared

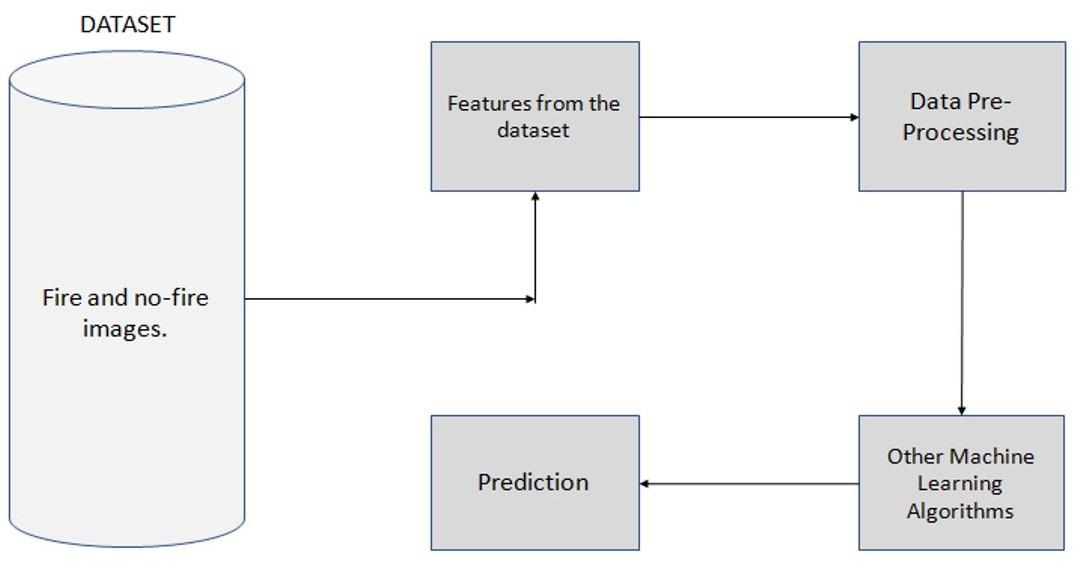
to other models such as AlexNet include its better classification accuracy, small sized model, and suitability of implementation on FPGAs and other hardware architectures having

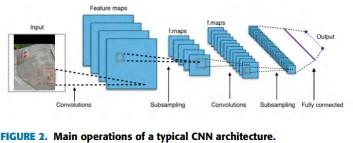
memory constraints. The intended architecture consists of

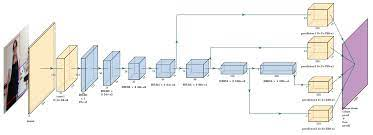
100 layers with 2 main convolutions, 4 max pooling, one average pooling, and 7 inception modules

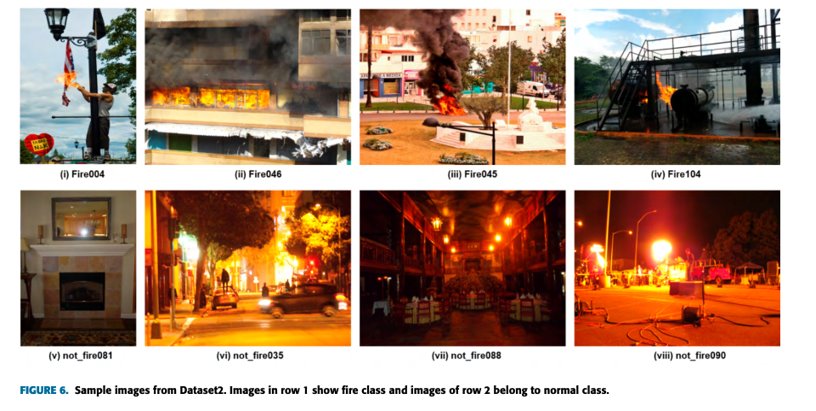


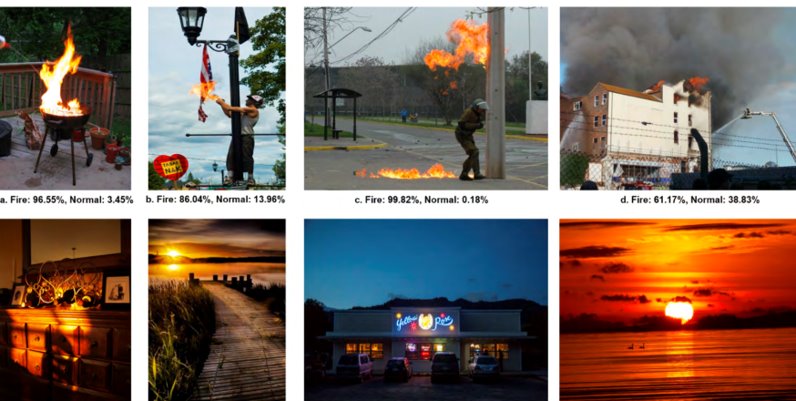
* 1. 3.1 BLOCK DIAGRAM:





* 1. 3.2 ALGORITHM:
  3. MobileNet is TensorFlow’s first mobile computer vision model.
  4. It uses depthwise separable [convolutions](https://builtin.com/machine-learning/fully-connected-layer) to significantly reduce the number of parameters compared to other networks with regular convolutions and the same depth in the nets. This results in lightweight [deep neural networks](https://builtin.com/machine-learning/what-is-deep-learning). A depthwise separable convMobileNet is a class of [convolutional neural network](https://builtin.com/data-science/convolutional-neural-networks-explained) (CNN) that was open-sourced by Google, and therefore, provides an excellent starting point for training classifiers that are insanely small and insanely fast.
  5. The speed and power consumption of the network is proportional to the number of multiply-accumulates (MACs) which is a measure of the number of fused multiplication and addition operations.solution is made from two operations - Depthwise and Pointwise convolution.
  6. 
  7. Dataset Specification
  8. Dataset1 is collected by Foggia et al. [14], containing 31 videos which cover different environments. This dataset has 14 fire videos and 17 normal videos without fire. The dataset is challenging as well as larger in size, making it a better option for experiments. The dataset has been madechallenging for both color-based and motion-based fire detection methods by capturing videos of fire-like objects and mountains with smoke and clouds. This is one of the motivations for selection of this dataset for our experiments.
  9. Dataset2 was obtained from [30], containing 226 images out of which 119 images belong to fire class and 107 images belong to non-fire class. The dataset is small but very challenging as it contains red- colored and fire-colored objects, fire-like sunlight scenarios, and fire- colored lightings in different buildings





* 1. Image Augmentation and Annotation

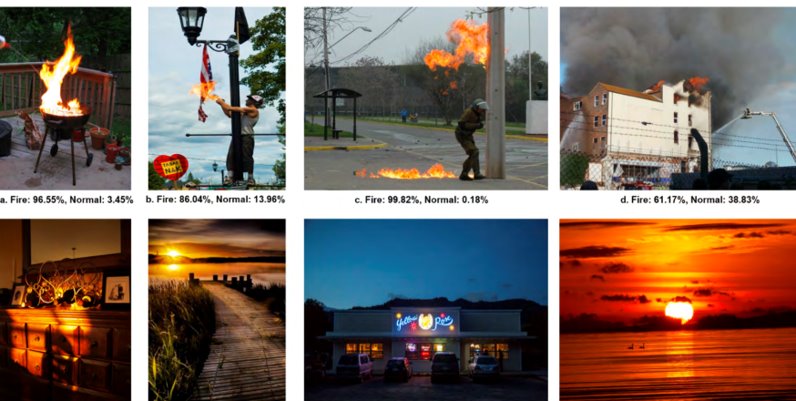
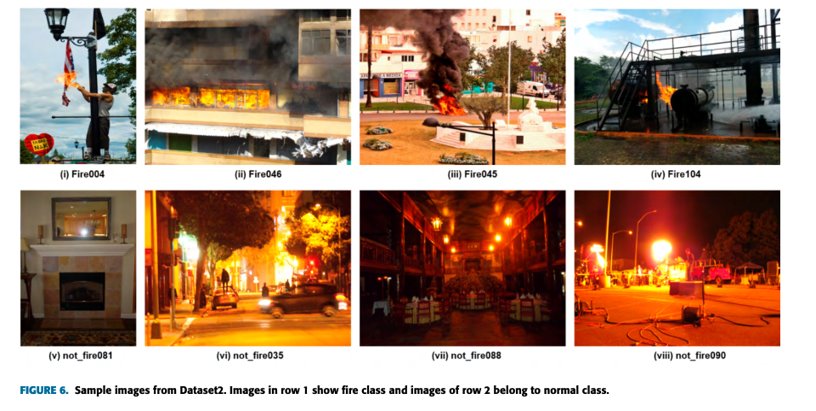
The results are compared with other flame detection methods, which are carefully selected using a selection criteria, reflecting the features used for fire detection, time, and dataset. The best results are reported by [14] among the existing recent methods by achieving an accuracy of 93.55% with 11.67% false alarms. The score of false alarms is still high and needs further improvement. Therefore, we explored deep learning architectures (AlexNet and GoogleNet) for this purpose. The results of AlexNet for fire detection are taken from our recent work [2]. Initially, we trained GoogleNet model with its default kernel weights which resulted in an accuracy of 88.41% with false positives score of 0.11% The baseline GoogleNet architecture randomly initializes the kernel weights which are tuned according to the accuracy and error rate during the training process. In an attempt to improve the accuracy, we explored transfer learning [33] by initializing the weights from pre-trained GoogleNet model and kernel the learning rate threshold to 0.001. Further, we also changed the last fully connected layer as per the nature of the intended problem. With this fine-tuning process, we reduced the false alarms rate from 0.11% to 0.054% and false negatives score from 5.5% to 1.5%, respectively.

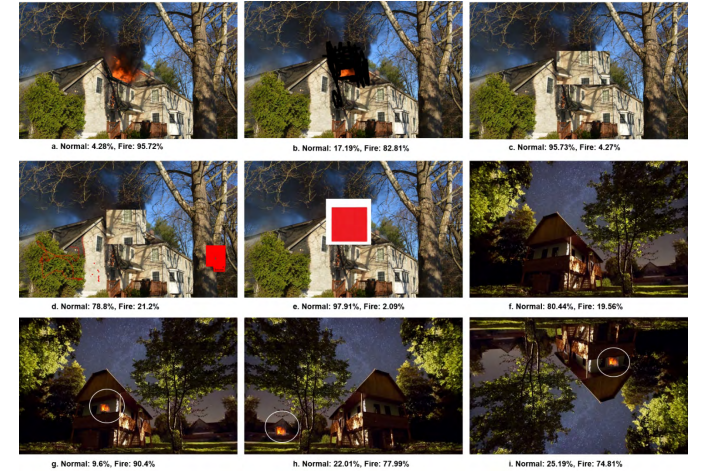
**CHAPTER 4: EXPERIMENTAL STUDY**

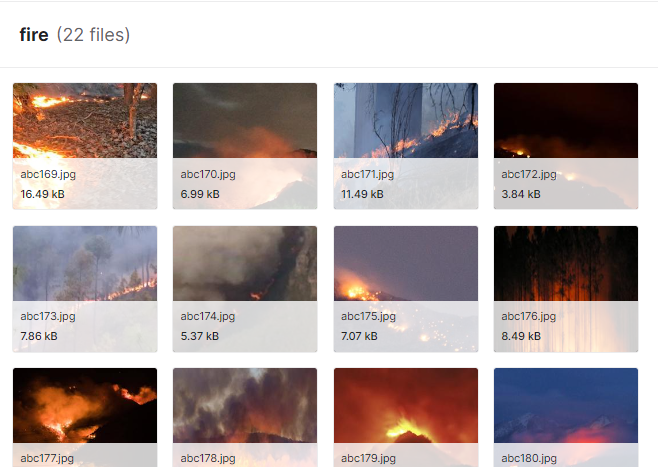
4.1 Data sets :

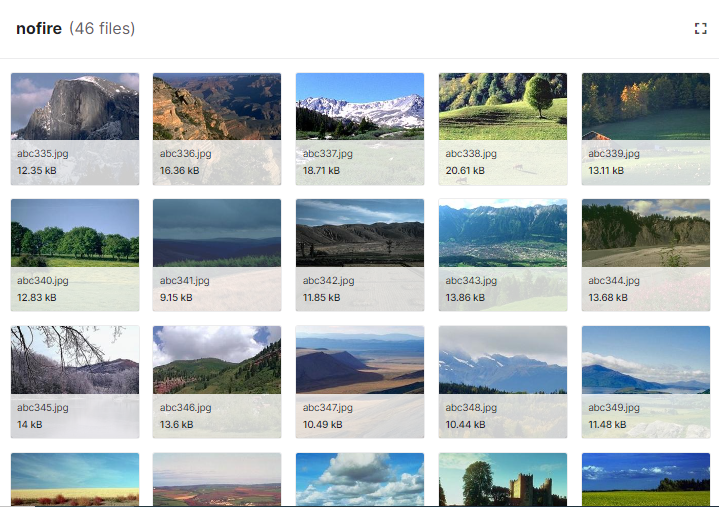
Open-source link:

* [https://www.kaggle.com/datasets/brsdincer/wildfire-detection-image-data​](https://www.kaggle.com/datasets/brsdincer/wildfire-detection-image-data%E2%80%8B)
* Foggia dataset.









We have taken images from different sources and combines them to make a customized dataset which deals with every challenging aspect of fire detection in every possible kind of fire prone environment.

4.2 Software Requirements :

* Windows 11 OS.
* Deep Learning models from specific library
* Visual Studio Code to analyze the results from the pre-trained models on some crop.
* Google collab
* Technologies:

-Tensorflow

-Keras

-Matplotlib

-Numpy

4.3 Hardware Requirements

* Dell Intel core i5 PC
* User Requirements :

Good understanding of fire detection principles.

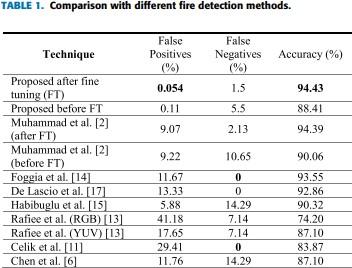
### 4.6 : RESULTS:

**CNN Results:**

For the deep learning models, we have used accuracy, precision, recall and F1 score as metrics to validate the best result generating model.

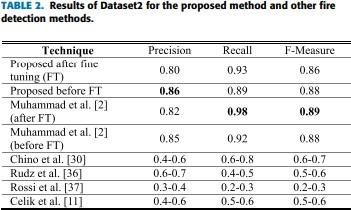
Accuracy is the total number of corrects to the total dataset.

The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.

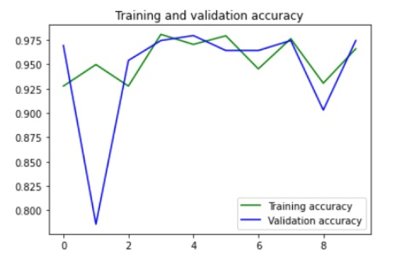


**Precision** is calculated by dividing the true positives by anything that was predicted as a positive. The number of instances that are relevant**,** out of the total instances the model retrieved.

**Recall** (or True Positive Rate) is calculated by dividing the true positives by anything that should have been predicted as positive. The number of instances which the model correctly identified as relevant out of the total relevant instances**.**



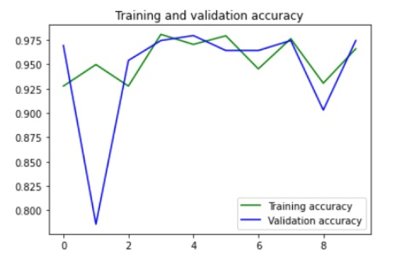
**Mobilenet results:**

****

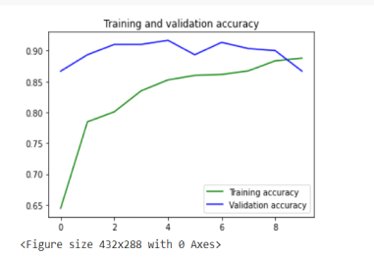
****

**4.7 ANALYSIS:**

**Mobilenet accuracy:**

****

**CNN accuracy:**



**CHAPTER 5: SUMMARY & FUTURE SCOPE**

Our application can be improved by utilizing a bigger dataset of fires at different stages and proportions to train the model.​

* With more significant GPU capability, we may deploy two deep learning models for characteristic examination, with their results feature vectors connected and categorized for improved robustness​
* Model can be embedded to a IoT device like Rasberry pi, Arduino to alert surroundings.​

The future of the global fire detector market looks promising with opportunities in the commercial, public institution, industrial, and residential sectors. The global fire detector market is forecast to reach $8.1 billion by 2027 with a CAGR of 6.5% from 2021 to 2027. The major drivers for this market are growth in the construction industry and stringent government regulations for fire safety.

The first detectors were for the detection of heat, and as time and technology advanced, they were also used for fixed temperature, rate-of-rise, rate anticipation and linear. These detectors are still in use today and for a number of applications remain a viable means of detection, though not for the purpose of life safety.

Through the use of thermistors and the software/firmware of the detector and the system, I do see that the Response Time Index (RTI) of a heat detector can be reduced so that the detection of a thermal event could be more quickly detected.

Fire alarm systems, however, are designed and installed in the majority of applications for life safety. The only detector that is used for this application is the smoke detector. Smoke detectors and smoke alarms are and remain as the single best method for the early detection of a fire and have saved countless lives. These devices however have one principle problem, they are a source for unwanted alarms.

Since the first generation of [**smoke detectors**](https://www.securitysales.com/tag/smokedetection) were released, there have been a number of advancements to both decrease the time of detection while at the same time decrease the activation of the detector when the products of combustion are not present. Smoke detectors and alarms are migrating from just the detection of smoke, to combination detectors and multicriteria detectors.

The future will be with multicriteria detection in which the detector will be more of a sensor, with the detection more for the products of combustion, such as carbon monoxide, carbon dioxide, sulfur dioxide, nitrogen oxides in addition to heat and particulate matter.

Sensors will also have the ability to sense or track when a room is occupied or not and have the ability to be integrated with occupant notification and evacuation. The development of more advanced algorithms and [**artificial intelligence**](https://www.securitysales.com/tag/artificial-intelligence), both within the sensor itself and the front end control unit will decrease the time from the beginning of an event to the notification of the event.

It is not improbable that detection technology will be able to detect an incipient fire at that stage rather than at the flaming stage. This at the same time could reduce the likelihood of an unwanted activation from occurring.

Within the next decade, video image detection (VID) will become more mainstream in which, through analytics, the image of either smoke or flame will be able to be isolated and detected from within a room or space. The VID system would also be able to detect if an individual is within the space and through the integration with the notification appliances, provide a path of exit.

In regard to the notification to the occupants, within the United States we are still primarily sounding an alarm throughout the occupancy and trusting that the occupants will heed the warning and head to the nearest exit.This does take time away from suppression and does increase the risks to the first responders that are conducting the searches. Through the use of sensors and VID, the location of an individual or individuals could be detected by the system and relayed to the first responders so that they could go directly to where they are located.

### REFERENCES

[1] K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H. H. Ge Wang, and S. W. Baik, ‘‘secure surveillance framework for IoT systems using probabilistic image encryption,’’ IEEE Trans. Ind. Inform., to be published, doi: <https://doi.org/10.1109/TII.2018.2791944>

[2] K. Muhammad, J. Ahmad, and S. W. Baik, ‘‘Early fire detection using convolutional neural networks during surveillance for effective disaster management,’’ Neurocomputing, vol. 288, pp. 30–42, May 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/ S0925231217319203

[3] J. Choi and J. Y. Choi, ‘‘An integrated framework for 24-hours fire detection,’’ in Proc. Eur. Conf. Comput. Vis., 2016, pp. 463–479.

[4] H. J. G. Haynes. (2016). Fire Loss in the United States During 2015. [Online]. Available: <http://www.nfpa.org/>

[5] C.-B. Liu and N. Ahuja, ‘‘Vision based fire detection,’’ in Proc. 17th Int. Conf. Pattern Recognit. (ICPR), Aug. 2004, pp. 134–137.

[6] T.-H. Chen, P.-H. Wu, and Y.-C. Chiou, ‘‘An early fire-detection method based on image processing,’’ in Proc. Int. Conf. Image Process. (ICIP), Oct. 2004, pp. 1707–1710.

[7] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Çetin, ‘‘Computer vision based method for real-time fire and flame detection,’’ Pattern Recognit. Lett., vol. 27, pp. 49–58, Jan. 2006.

[8] J. Choi and J. Y. Choi, ‘‘Patch-based fire detection with online outlier learning,’’ in Proc. 12th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), Aug. 2015, pp. 1–6.

[9] G. Marbach, M. Loepfe, and T. Brupbacher, ‘‘An image processing technique for fire detection in video images,’’ Fire Safety J., vol. 41, no. 4, pp. 285–289, 2006.

[10] D. Han and B. Lee, ‘‘Development of early tunnel fire detection algorithm using the image processing,’’ in Proc. Int. Symp. Vis. Comput., 2006, pp. 39–48.

[11] T. Çelik and H. Demirel, ‘‘Fire detection in video sequences using a generic color model,’’ Fire Safety J., vol. 44, no. 2, pp. 147–158, 2009.

[12] P. V. K. Borges and E. Izquierdo, ‘‘A Probabilistic approach for visionbased fire detection in videos,’’ IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 5, pp. 721–731, May 2010.

[13] A. Rafiee, R. Dianat, M. Jamshidi, R. Tavakoli, and S. Abbaspour, ‘‘Fire and smoke detection using wavelet analysis and disorder characteristics,’’ in Proc. 3rd Int. Conf. Comput. Res. Develop. (ICCRD), Mar. 2011, pp. 262–265.

[14] P. Foggia, A. Saggese, and M. Vento, ‘‘Real-time fire detection for videosurveillance applications using a combination of experts based on color, shape, and motion,’’ IEEE Trans. Circuits Syst. Video Technol., vol. 25, no. 9, pp. 1545–1556, Sep. 2015.

[15] Y. H. Habiboğlu, O. Günay, and A. E. Çetin, ‘‘Covariance matrix-based fire and flame detection method in video,’’ Mach. Vis. Appl., vol. 23, no. 6, pp. 1103–1113, 2012.

[16] M. Mueller, P. Karasev, I. Kolesov, and A. Tannenbaum, ‘‘Optical flow estimation for flame detection in videos,’’ IEEE Trans. Image Process., vol. 22, no. 7, pp. 2786–2797, Jul. 2013.

**APPENDIX:**

**LINKS:**

**CODE:**

**CNN CODE:**

# Import Libraries

import pandas as pd

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras import datasets, layers, models

from tensorflow.keras.layers import (

BatchNormalization, Conv2D, MaxPooling2D, Flatten, Dropout, Dense

)

import matplotlib.pyplot as plt

import numpy as np

import os

import glob

from tqdm import tqdm

import cv2

import sklearn

import skimage

from skimage.transform import resize

import random

from keras.preprocessing.image import ImageDataGenerator

from sklearn.metrics import accuracy\_score

import seaborn as sns

sns.set()

train = "../input/wildfire-detection-image-data/forest\_fire/Training and Validation"

test = "../input/wildfire-detection-image-data/forest\_fire/Testing"

LOAD\_FROM\_IMAGES = True

def get\_data(folder):

x = []

y = []

for folderName in os.listdir(folder):

if not folderName.startswith("."):

if folderName in ["nofire"]:

label = 0

elif folderName in ["fire"]:

label = 1

else:

label = 2

for image\_filename in tqdm(os.listdir(folder +"/" +folderName+"/")):

img\_file = cv2.imread(folder + "/" +folderName + "/" + image\_filename)

if img\_file is not None:

img\_file = skimage.transform.resize(img\_file,(227,227,3), mode = "constant",anti\_aliasing=True)

img\_arr = np.asarray(img\_file)

x.append(img\_arr)

y.append(label)

x = np.asarray(x)

y = np.asarray(y)

return x,y

if LOAD\_FROM\_IMAGES:

X\_train,y\_train = get\_data(train)

X\_test, y\_test = get\_data(test)

np.save("xtrain.npy",X\_train)

np.save("ytrain.npy",y\_train)

np.save("xtest.npy",X\_test)

np.save("ytest.npy",y\_test)

else:

X\_train = np.load("xtrain.npy")

y\_train = np.load("ytrain.npy")

X\_test = np.load("xtest.npy")

y\_test = np.load("ytest.npy")

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

# Split the data

X\_train, X\_valid, y\_train, y\_valid = train\_test\_split(X\_train,y\_train,test\_size=0.2,shuffle=True)

# Image Normalization

X\_train, X\_valid, X\_test = X\_train / 255.0, X\_valid / 255.0, X\_test / 255.0

model = models.Sequential()

model.add(layers.Conv2D(96,(11,11),strides=(4, 4),activation="relu",input\_shape=(227,227,3)))

model.add(BatchNormalization())

model.add(layers.MaxPooling2D((3,3), strides=(2,2)))

model.add(layers.Conv2D(256,(5,5),activation="relu",padding="same"))

model.add(BatchNormalization())

model.add(layers.MaxPooling2D((3,3), strides=(2,2)))

model.add(layers.Conv2D(384,(3,3),activation="relu",padding="same"))

model.add(BatchNormalization())

model.add(layers.Conv2D(384,(3,3),activation="relu",padding="same"))

model.add(BatchNormalization())

model.add(layers.Conv2D(256,(3,3),activation="relu",padding="same"))

model.add(BatchNormalization())

model.add(layers.MaxPooling2D((3,3), strides=(2,2)))

model.add(layers.Flatten())

# Fully connected

model.add(layers.Dense(4096,activation="relu"))

model.add(Dropout(0.5))

model.add(layers.Dense(4096,activation="relu"))

model.add(Dropout(0.5))

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

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

from tensorflow.keras.callbacks import EarlyStopping

early\_stopping = EarlyStopping(

monitor="accuracy",

patience=5,

restore\_best\_weights=True)

batch\_size=8

epochs=100

history = model.fit(X\_train,y\_train,validation\_data=(X\_valid,y\_valid),batch\_size=batch\_size,

epochs=epochs,verbose=1,callbacks=[early\_stopping])

score = model.evaluate(X\_test, y\_test, batch\_size=batch\_size, verbose=1)

print('Test loss:', score[0])

print('Test accuracy:', score[1])

from matplotlib import pyplot as plt

%matplotlib inline

plt.figure(figsize=(6, 5))

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

plt.plot(history.history['val\_accuracy'], color='b')

plt.title('Model Accuracy', weight='bold', fontsize=16)

plt.ylabel('accuracy', weight='bold', fontsize=14)

plt.xlabel('epoch', weight='bold', fontsize=14)

plt.ylim(0.6, 1.0)

plt.xticks(weight='bold', fontsize=12)

plt.yticks(weight='bold', fontsize=12)

plt.legend(['train', 'val'], loc='upper left', prop={'size': 14})

plt.grid(color = 'y', linewidth='0.5')

plt.show()

y\_test\_pred = model.predict(X\_test)

y\_pred = (y\_test\_pred > 0.5)

from sklearn.metrics import classification\_report

from sklearn.metrics import confusion\_matrix

y\_pred.shape

y\_test.shape

print(classification\_report(y\_test, y\_pred))

def confusion(y\_test,y\_test\_pred,X):

names=['No Fire','Fire']

cm=confusion\_matrix(y\_test,y\_test\_pred)

f,ax=plt.subplots(figsize=(10,10))

sns.heatmap(cm,annot=True,linewidth=.5,linecolor="r",fmt=".0f",ax=ax)

plt.title(X, size = 25)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

ax.set\_xticklabels(names)

ax.set\_yticklabels(names)

plt.show()

return

confusion(y\_test,y\_pred,"CNN")

model.save("wildfire\_model.h5")

**MOBILENET CODE:**

**from google.colab import drive**

**drive.mount("/content/gdrive")**

**dataset = "/content/gdrive/MyDrive/fire\_dataset"**

**import os**

**os.listdir("/content/gdrive/MyDrive/fire\_dataset/train")**

**#importing all the necessary libraries**

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**%matplotlib inline**

**import cv2**

**import tensorflow**

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

**from sklearn.metrics import classification\_report,confusion\_matrix**

**img\_path='/content/gdrive/MyDrive/fire\_dataset/train/fire\_images/fire\_0465.jpg'**

**img=cv2.imread(img\_path)**

**img.shape**

**plt.imshow(img)**

**classes=['fire\_images', 'non\_fire\_images']**

**img\_size=224**

**training\_Data=[]**

**def create\_training\_Data():**

**for category in classes:**

**path = os.path.join(dataset,'train',category)**

**class\_num=classes.index(category)**

**for img in os.listdir(path):**

**try:**

**img\_array=cv2.imread(os.path.join(path,img),cv2.IMREAD\_GRAYSCALE)**

**backtorgb=cv2.cvtColor(img\_array,cv2.COLOR\_GRAY2RGB)**

**new\_array=cv2.resize(backtorgb,(img\_size,img\_size))**

**training\_Data.append([new\_array,class\_num])**

**except Exception as e:**

**pass**

**create\_training\_Data()**

**print(len(training\_Data))**

**import random**

**random.shuffle(training\_Data)**

**X=[]**

**y=[]**

**for features,label in training\_Data:**

**X.append(features)**

**y.append(label)**

**trainX=np.array(X).reshape(-1,img\_size,img\_size,3)**

**trainY=np.array(y)**

**trainX.shape**

**test\_Data=[]**

**def create\_test\_Data():**

**for category in classes:**

**path = os.path.join(dataset,'test',category)**

**class\_num=classes.index(category)**

**for img in os.listdir(path):**

**try:**

**img\_array=cv2.imread(os.path.join(path,img),cv2.IMREAD\_GRAYSCALE)**

**backtorgb=cv2.cvtColor(img\_array,cv2.COLOR\_GRAY2RGB)**

**new\_array=cv2.resize(backtorgb,(img\_size,img\_size))**

**test\_Data.append([new\_array,class\_num])**

**except Exception as e:**

**pass**

**create\_test\_Data()**

**print(len(test\_Data))**

**X=[]**

**y=[]**

**for features,label in test\_Data:**

**X.append(features)**

**y.append(label)**

**testX=np.array(X).reshape(-1,img\_size,img\_size,3)**

**testY=np.array(y)**

**import tensorflow as tf**

**from tensorflow.keras.optimizers import RMSprop,Adam**

**model = tf.keras.models.Sequential([**

**tf.keras.layers.Conv2D(96, (11,11), strides=(4,4), activation='relu', input\_shape=(224, 224, 3)),**

**tf.keras.layers.MaxPooling2D(pool\_size = (3,3), strides=(2,2)),**

**tf.keras.layers.Conv2D(256, (5,5), activation='relu'),**

**tf.keras.layers.MaxPooling2D(pool\_size = (3,3), strides=(2,2)),**

**tf.keras.layers.Conv2D(384, (5,5), activation='relu'),**

**tf.keras.layers.MaxPooling2D(pool\_size = (3,3), strides=(2,2)),**

**tf.keras.layers.Flatten(),**

**tf.keras.layers.Dropout(0.2),**

**tf.keras.layers.Dense(2048, activation='relu'),**

**tf.keras.layers.Dropout(0.25),**

**tf.keras.layers.Dense(1024, activation='relu'),**

**tf.keras.layers.Dropout(0.2),**

**tf.keras.layers.Dense(2, activation='softmax')**

**])**

**model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True),**

**optimizer=Adam(learning\_rate=0.0001),**

**metrics=['accuracy'])**

**model.summary()**

**history = model.fit(trainX,trainY,epochs=10,validation\_data=(testX, testY))**

**print(testX)**

**%matplotlib inline**

**import matplotlib.pyplot as plt**

**acc = history.history['accuracy']**

**val\_acc = history.history['val\_accuracy']**

**loss = history.history['loss']**

**val\_loss = history.history['val\_loss']**

**epochs = range(len(acc))**

**plt.plot(epochs, acc, 'g', label='Training accuracy')**

**plt.plot(epochs, val\_acc, 'b', label='Validation accuracy')**

**plt.title('Training and validation accuracy')**

**plt.legend(loc=0)**

**plt.figure()**

**plt.show()**

**plt.plot(epochs, loss, 'r', label='Training loss')**

**plt.plot(epochs, val\_loss, 'orange', label='Validation loss')**

**plt.title('Training and validation loss')**

**plt.legend(loc=0)**

**plt.figure()**

**plt.show()**

**print(val\_acc)**