Early Flame Detection System Using Real-Time Machine-Vision and Image Analysis

by

Tarek Ghareeb Mohamed

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Civil Engineering

Waterloo, Ontario, Canada, 2021

Author’s Declaration

Abstract

From 2010 to 2019, 110,811 fires with losses have been reported to the Office of Fire Marshall and Emergency Management in Ontario. In The USA, the local fire departments responded to 1,338,500 fires in 2020. These fires caused 3,500 civilian deaths, 15,200 civilian injuries and $21.9 billion in property damage. A fire occurs in a structure in the USA every 64 seconds. Those and similar recent statistics from different parts of the world indicate that the current point-type fire detection technology has failed to eliminate the hazards of death, injury, and economic loss caused by fire. This research aims to utilize the latest digital video processing and computer vision technology to develop a more efficient flame detection system.

Due to rapid developments in digital Cameras, IoT, and 5G telecommunication technologies, computer-vision based fire detection is getting more attention from researchers in recent years. Computer-vision based fire detection can be as simple as a single IoT camera to detect a fire early before becoming out of control and turning into a threatening risk, triggers a local alarm, and sends remote warning signals to the fire department and emergency management officials. The proposed system does not require high capital costs nor high operation and maintenance costs since it will run on top of the existing infrastructure of the digital security & surveillance system network. Moreover, the proposed system has broad potential for indoor and outdoor applications in urban areas, and it is easily expandable by adding more IP cameras to the existing network.

The proposed system incorporates two stages: Stage-I: Detecting the fire candidate region from live video stream based on colour and motion information; and Stage-II: passing the candidate region to a trained Convolutional Neural Network (CNN) classification model to classify the region as fire or non-fire. The main innovation in this approach is its simplicity and suitability for real-time use without compromising the accuracy. The experimental results show that the system training and validation accuracies reach 100% and 98% respectively.

Applying the proposed framework as an additional layer of protection integrated into existing indoor and outdoor digital security & surveillance systems is expected to provide early fire detection and allows firefighters and rescue teams to arrive at the scene at its early and offer priceless minutes to the attendees or building occupants to evacuate the hazardous locations. This proposal will save lives and minimize the economic loss in public and private properties.

Acknowledgements

I like to express my sincere appreciation and gratitude to my supervisor, Dr. Tarek Hegazy. His invaluable guidance and his wise notes have enriched my research. I am deeply impressed with his excellence in teaching and dedication to academic research. It has been a wonderful opportunity to learn from his experience. Special thanks to the committee members: Dr. Chul Min Yeum and Dr. Dipanjan Basu, for their insightful comments.

I am so grateful to my parents, who taught me how to learn. My father, May God shower him with his blessing and mercy, and my mother, may God give her a happy and healthy life. And the last but not least, I like to express my warm thanks to my family, my wife, Amany, and the kids: Sama, Hana, and Adam, for their endless love and support.

Table of Contents

List of Figures

Chapter 1. Introduction

1.1. General

In Canada, fires cause extensive yearly losses in property and even human lives. From 2010 to 2019, there were 110,811 fires with loss reported to the Office of Fire Marshall and Emergency Management in Ontario [21] (Figure 1.1, Figure 1.2, and Figure 1.3). Similar fire incidents have been reported in other parts of Canada (Figure 1.4).

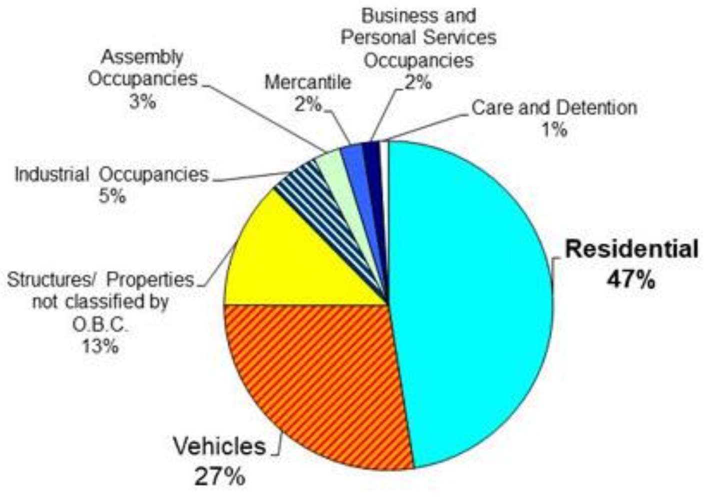


Figure 1.1 Fire Losses in Ontario from 2009 to 2018 by Property Class. [21]

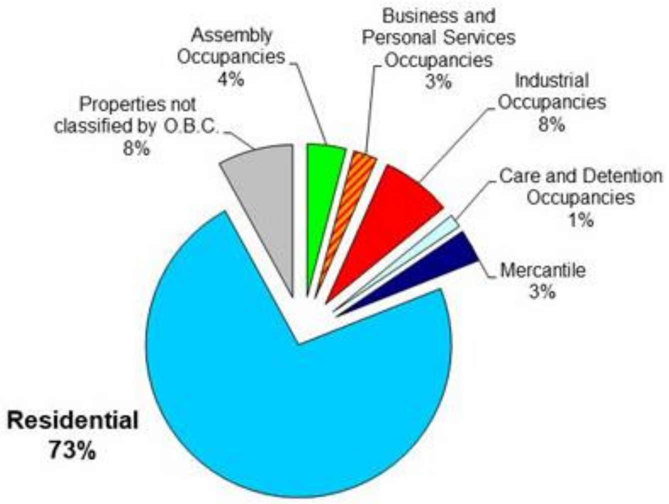


Figure 1.2 Fire Losses in Ontario from 2009 to 2018 by Property Class – Structure only. [21]

Figure 1.3 Massive fire breaks out in townhouse complex in Toronto, December 2020. [5]

Figure 1.4 Massive fire on property of Headingley, Manitoba, December 2020. [3]

In The USA, the local fire departments responded to 1,338,500 fires in 2020. These fires caused 3,500 civilian deaths, 15,200 civilian injuries and $21.9 billion in property damage. A fire occurs in a structure at the rate of one every 64 seconds. [19], Figure 1.5

Figure 1.5 Massive fire in buildings under construction Denver, USA, May 2018. [33]

The large number of fire incidents are happening while the local building codes and standards in both Canada and the US mandate the use of fire detection and fire fighting systems in almost every structure. The above statistics reveal that the existing conventional point-type smoke alarms and other fire detection systems are not sufficient to eliminate the hazards of death, injury, and economic loss caused by fire accidents. The conventional point-type fire detection technology suffers from the following limitations:

1. **Slow:** it takes up to minutes for the accumulated smoke and/or heat to trigger the sensors.
2. **Limited coverage:** the point-type fire detectors have limited coverage. For example, a typical smoke detector covers a circular range of 6.4 m. (21 ft.) radius according to NFPA 72 and CAN ULC S524, which makes it less effective in wide and large spaces with high ceilings.
3. **Expensive:** high number of detectors are required to cover large buildings. Those detectors have to be connected to a central control panel using cables, which increases both the installation and the maintenance costs.

4. **Not flexible:** upgrading the point-type fire detection and alarm system because of building renovations and additions is costly. In most cases, the entire system and the main fire control panel have to be replaced to suit the building upgrades**.**
5. **Indoor only:** the point-type fire detectors are inapplicable in the outdoor environment.

For example, large open structures such as construction sites, power stations, oil and gas facilities, wastewater treatment plants, tunnels, aircraft hangars, storage warehouses, parking garages and construction sites are highly vulnerable to fire hazards. Mainly because of the daily ongoing activities that can easily cause a fire. Furthermore, the lack of enough fire-rated barriers installed. So that, once happening, fire can spread out freely and cause severe consequences. Another example, in case of ground fire in high ceiling assembly occupancies such as auditoriums, atriums, railway stations, etc., smoke or heat will take a long time to rise upwards and trigger the ceiling-mounted conventional heat or smoke detectors. So, it will be too late when the conventional sensors detect fire conditions. By then, the fire would have spread out in space already and caused severe economic losses, and even worst, potential injuries and fatalities. It is concluded from the above two examples that whenever the fire detection system is most needed, the conventional point-type fire detection system is least effective or totally inapplicable.

Therefore, there is a need for a novel fire detection technology that can be used efficiently for indoor and outdoor applications. The technology that is capable of early fire detection and providing essential minutes to the firefighters and rescue teams to arrive at the scene to fight the fire at its early stage and give priceless time for the attendees, the labor, or the building occupants to evacuate the hazardous locations. Here, computer-vision based fire detection comes into the picture as an effective solution to fill the gap where the conventional system is least efficient or inapplicable.

Besides the early fire detection, the computer-vision based system is capable to provide more information on the direction, size, and growth of the fire region. So that the system can track how fast the fire is spreading and send this critical information to the fire department, giving the firefighters a clearer understanding of the situation even before they arrive at the fire scene. The high-speed G5 telecom networks will facilitate fast data transfer from remote areas to the nearest fire department and police station.

1.2. Research Motivation

This research aims to use the latest technology in digital video processing and computer-vision algorithms to develop a new real-time computer-vision based flame detection system. The proposed method can automatically detect a flame of fire at a very early stage with high accuracy using the standard IP surveillance camera systems (Figure 1.6), which are widely used almost anywhere: indoor, and outdoor. The research has been motivated by the following:

Figure 1.6 Surveillance cameras are used everywhere, Indoor & Outdoor.

1.2.1. Need for Fast-Acting Fire Detection System

The point-type fire detection system is less efficient in high ceiling structures such as arenas, atriums, auditoriums, theatres, warehouses, hangars, etc. The buildup of smoke and/or heat has to travel upward from the location of fire up to the ceiling-mounted smoke and/or heat detectors, activate the sensors, and trigger the alarm. A standard IP camera can capture a video with 480p or 720p resolution at 30 frames per second (fps).

Knowing that a small flame needs around 30 seconds to develop a large fire [26], during this time, the surveillance camera can capture 30X30=900 successive frames (images) before the developing fire become completely out of control and turn into a threatening risk. This amount of information is more than sufficient to detect the fire condition in the scene and trigger the fire alarms fast enough to avoid life and property losses. Modern surveillance cameras capture video at higher resolution (full high definition, FHD), which improves fire detection capability.

1.2.2. Need for Efficient and Wide Coverage System for Indoor and Outdoor

As mentioned earlier, the conventional point-type fire detectors system, which has been used for many years and is still in use, suffers from two main disadvantages: first, it has limited coverage. So that for large structures, many detectors distributed all over the entire space are required for optimum coverage. The detectors should be interconnected to a central control panel through signaling cables. Second, the point-type fire detectors are inefficient in outdoor applications because the weather conditions such as wind, temperature, humidity and rain hinder the sensitivity of the sensors.

In contrast, the same large structures and surroundings are monitored with a few IP surveillance cameras located at strategic indoor and outdoor points. The same cameras can be utilized as early and accurate fire detectors. A single-camera can cover a space that would require many conventional point-type detectors to cover. The camera installed on a high ceiling can accurately and quickly capture the point of fire on the ground and trigger the alarm early enough, allowing the evacuation of the attendee and occupants from the fire scene.

In addition, a computer-vision based fire detection system is suitable for outdoor applications, as it is less sensitive to weather conditions. These two advantages of the computer-vision based fire detection over the conventional point-type fire detection (the efficiency and the coverage) make it ideal for fire detection applications in high ceiling structures and outdoor such as parking lots, open warehouses, construction sites, arenas, atriums, auditoriums, theatres, railway stations, aircraft hangars, to name a few.

1.2.3. Need for Integrated, and Expandable System for Life Safety and Security

Combining both security cameras and fire detection systems into one integrated digital life safety and security system will lead to functionality improvement. Furthermore, the integrated system will have internet connectivity, which allows for remote monitoring and alarms. In addition, the integrated system is highly flexible for building renovations and alterations, and easily expandable for building additions by adding more IP cameras to the central head end equipment, (Figure 1.7).

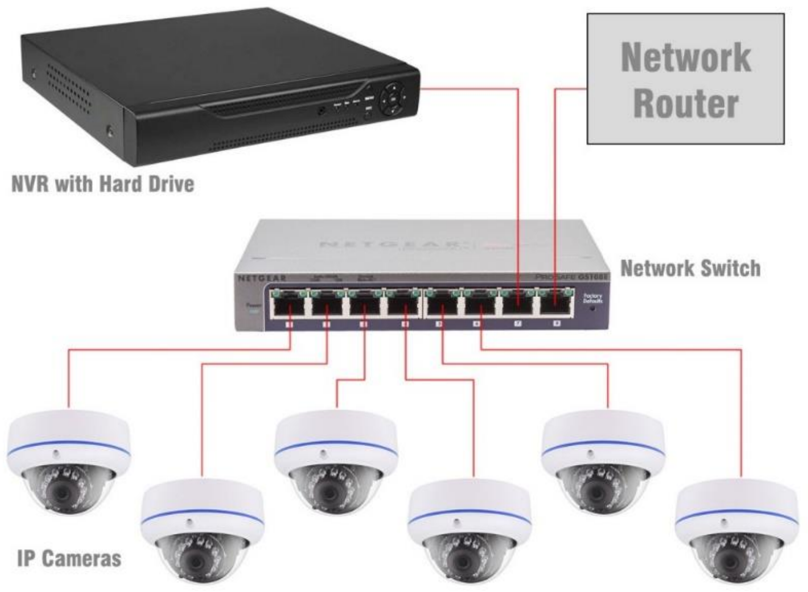


Figure 1.7 IP cameras network allows for more flexibility and expandability.

1.2.4. Need for A Low-Cost Fire Detection System

The need for a computer-vision based fire detection system that will run on top of the existing infrastructure of the digital security & surveillance system network. Such a system will improve reliability, maintainability and reduce the initial installation cost and the operation and maintenance costs.

1.3. Research Objectives and Scope

This research framework aims to build a prototype real-time computer-vision based flame detection system for indoor and outdoor applications. The system that is capable of filling the gap where the conventional point-type fire detection system fails to detect fire conditions or has a late response. The proposed system will be capable to detect fire incidents at high accuracy as early as possible before becoming totally out of control and turning into a threatening risk. The proposed system is an integrated solution that combines surveillance and early fire detection in one system. The system is flexible and scalable to fit almost any building size and application. Furthermore, the proposed system will be suitable for outdoor applications. Detailed research objectives are as follows:

1. Investigate the visual features of the flames caused by burning common combustible materials and identify the research gaps in the existing literature of fire detection;
2. Collect a large number of videos and images, with and without fires, from online image libraries and online search tools like Google images. Accordingly utilize the collected data to develop fire region detection algorithm based on image processing. This algorithm has to be simple enough to run on a low-cost hardware without compromising the high accuracy;
3. Build a prototype real-time fire detection system utilizing a CNN-based image classification algorithm that is modelled using the Python OpenCV computer programming language and the deep learning library Keras/TensorFlow. The model involves optimizing the CNN hyper-parameters for better learning process; and
4. Experiment with Google Colab platform (https://colab.research.google.com/) to efficiently train the CNN model for image classification. Accordingly, test and validate the performance of the proposed algorithm on real fire videos and images as well as other non-fire images with fire-coloured objects. Consequently, discuss the prospect of utilizing the proposed system for wide implementation to early detect and prevent fires.

1.4. Research Methodology

The methodology for achieving the above objectives is described below:

1. Conduct detailed literature review of the current methods of computer-vision based fire detection. The review focuses on four main areas of research: Fire detection based on image processing; Fire detection based on video processing; Fire detection based on advanced video processing; and Smoke detection;
2. Study the visual features of the flame caused by burning common combustible materials that can be found in the environments, such as wood, plastic, paper, or others. Accordingly, identify of the limitation of the current methods in the literature and the research gap;
3. Study the digital image processing algorithms for colour filters, background subtraction, edge detection, object segmentation, texture analysis, shape analysis, extraction of spatiotemporal statistical features, and image classification;
4. Study the traditional machine learning algorithms such as Support Vector Machine (SVM) and the new advances in machine learning algorithms for image processing such as Convolutional Neural Network (CNN);

5. Collect image and video Dataset for Model Training and Validation from the resources indicated in the reviewed literature. In addition, utilize online image libraries and search tools like Google-image to collect large nubmer of images representing real fire and fire-coloured objects;
6. Split the dataset into two subsets for training and validation purposes. Also, define a comprehensive list of performance metrics to measure the model performance during training and validation phases;
7. Develop a Novel Real-Time Fire Detection Model in two stages: (1) Detecting fire candidate region(s) from video stream captured by a surveillant camera in real-time; (2) examining the detected candidate regions by a trained CNN-based image classification algorithm to classify the region as real fire or non-fire object;
8. Code a Prototype Video Processing Model (Stage-I) using the OpenCV-Python open-source library for computer-vision programming to implement the proposed real-time model;
9. Code a Prototype CNN image Classification Model (Stage-II) using OpenCV-Python and the Keras library, which is a high-level deep learning library that runs on the top of the TensorFlow open-source low-level platform for artificial intelligence applications; and
10. Extensively test and measure the Model Performance to validate its performance.

1.5. Thesis Organization

The remainder of the thesis is organized as follows:

Chapter 2 presents a comprehensive literature review to determine the unique features of fire regions in real fire images and to understand the state-of-the-art computer-vision based fire detection algorithms. The review includes insight into the current trends in vision-based fire detection techniques. In addition, the capabilities and limitations of each technique are discussed.

Chapter 3 demonstrates the process of video and image data gathering and preparation, introduces in-depth review of the proposed 2-stages computer-vision based flame detection model. The main components and the system architecture are presented.

Chapter 4 discusses the prototype CNN model implementation and performance. Training and validation processes and various performance metrics including confusion matrix, precision, recall rate, and F-score have been covered in this chapter.

Chapter 5 describes summary of the research works, highlighting the research contributions, recommendations and provide suggestions for future research.

Chapter 2. Literature Review

2.1. Introduction

This chapter starts with background information on the basic concepts and calculations related to digital image processing, then the visual features of the flame region (as described in the modern literature) are collected and demonstrated. The machine learning and deep learning methods and the difference between them are briefly discussed. The concept and the main components of the Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) models are presented. The rest of this chapter presents a comprehensive review of the recently developed computer-vision-based fire detection algorithms. The chapter ends by summarizing the research gaps that serve as the foundation to establish this research framework as in Chapter 3.

2.2. Background on Image Analysis

Digital image can be represented as two-dimensional array of M x N picture elements, also known as pixels, where M is the image width, which is the number of columns of the matrix, and N is the image height, which is the number of rows of the matrix, (Figure 2.1). Gray scale digital image is saved in computers as a function, f (m, n), the values of f (m, n) at each pixel is the brightness of the image at that pixel [10]. In the commonly used 8-bit grayscale system, the pixel intensity can take a value in 256 (2^8) pins scale ranging from 0 (represents black) up to 255 (represents white).

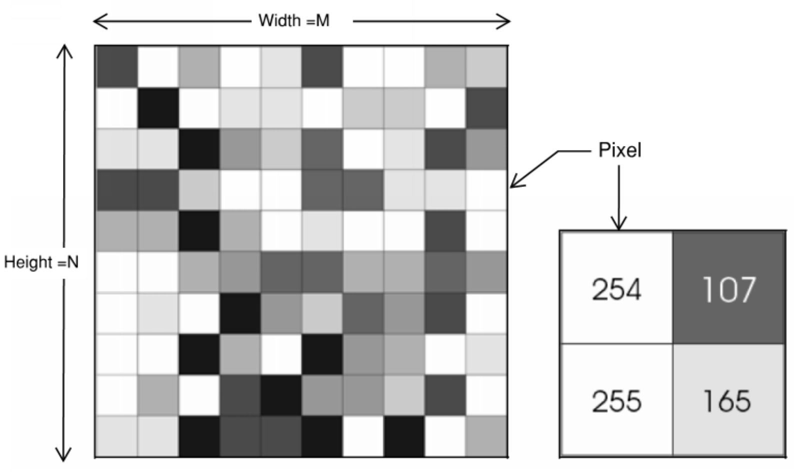


Figure 2.1: 8-bit grayscale digital image.

2.2.1. Digital Colour Image

Similarly, the digital colour image represented in the well-known Red Green Blue [R, G, B] colour space has three colour values per pixel instead of the one brightness value in the grayscale image. Again, colour value is digitized into 256 pins scale ranging from Black pixel represented by (R, G, B) = (0,0,0) up to White pixel represented by (R,G,B) = (255, 255, 255). Pure red colour is represented by (R, G, B) = (255, 0, 0) [24].

It is common in digital image processing to transform the image from one colour space to another to analyze specific colour properties. Other colour space representations are YCbCr, YUV, CIE LAB, and HSV, which are widely used by researchers in colour segmentation because those colour spaces have one main advantage over RGB colour space, they separate between the brightness value (luma) and the colour values (chroma) which simulates how human beings perceive colour, and makes colour segmentation easier and more efficient at different illumination conditions [4]. In YCbCr system, the luma component is denoted by ‘Y’ whereas the chrominance-blue and chrominance-red components are denoted by ‘Cb’ and ‘Cr’ respectively. The range of ‘Y’ is 16–235, whereas the range of Cb and Cr is 16−240. The transformation from RGB values to YCbCr values is given by Eq. 1, [29], as follows:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.256 & 0.504 & 0.098 \\ -0.148 & -0.292 & 0.441 \\ 0.441 & -0.369 & -0.071 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.. (1)$$

Similarly, the YUV model defines a colour space in terms of one luma component (Y) and two chrominance components, called U (blue projection) and V (red projection) respectively, YUV and YCbCr are fairly similar but they are different formats with different scale factors. Today, the term YUV is commonly used in the computer industry to describe file-formats that are encoded using YCbCr [23].

The CIE LAB colour space also referred to as L*a*b* is a colour space defined by the International Commission on Illumination (CIE) in 1976. The colour is represented as three values: $L*$ for perceptual lightness, and $a*$ and $b*$ for the four unique colours of human vision: red, green, blue, and yellow [7].

Likewise, HSV (also referred to as HSI) is another linear transformation from the RGB. HSV also has three components: H: the colour component, also known as hue, S: colour saturation component, also known as hue saturation or simply saturation, and V (I): the luma component, also known as the value or illuminance [44], Figure 2.2. The range of ‘H’ is 0°−360° where 0° represent the red colour, range of ‘S’ is 0−1 where lower values represent washed out colour and higher values represent vibrant colour. Finally, the range of ‘V’ is 0−1 where lower values represent dark pixel and higher values represent bright pixel.

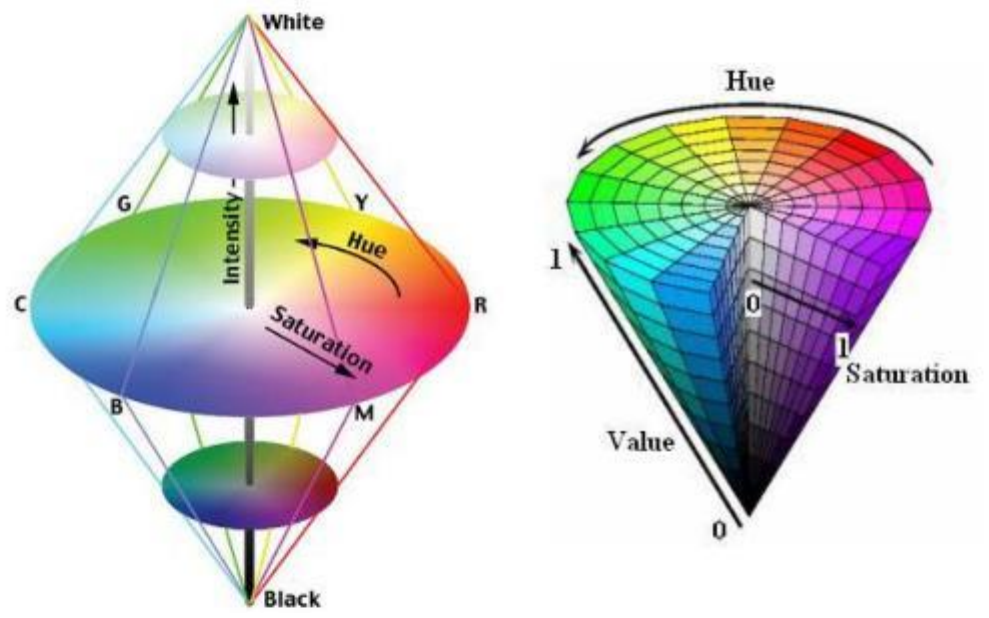


Figure 2.2 Visual representation for the HSV colour space.

2.2.2. Video Sequence

Video sequence is a series of discrete still images (frames) capturing motion of moving object at consecutive short moments, the human visual system can process 10 to 12 images per second and perceive them individually, while higher rates are perceived as continuous motion [25]. Standard modern digital surveillance cameras can capture videos at 30 frames/second.

Technically, there is no significant difference between digital images and digital videos, all the theories and algorithms that are applicable to images, can be applied to videos in the same manner. The only difference between digital image and digital video is that the video adds temporal information of the changes in the observed scene, that temporal information is analyzed to obtained valuable conclusions as it will detailed afterwards in this chapter.

2.2.3. Calculation of The Statistical Characteristics in Image Processing

The following statistical information are frequently used in the reviewed literature. It is helpful to list the calculation formulas:

- Mean value of the brightness (or colour components) in an image is given by Eq.2, [24], as follows: $$Mean = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} F(m,n) \quad \text{……………………... (2)}$$
- Brightness (colour) standard Deviation and Variance are given by Eq.3 and Eq.4, as follows:

$$Standard\ Deviation = \sqrt{\frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} |F(m,n) - Mean|^2} \quad \text{…….. (3)}$$

$$Variance = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} |F(m,n) - Mean|^2 \quad \text{…………………….. (4)}$$

- Area of the region of interest (ROI): in the context of image processing, the area of ROI means the count of the number of all pixels that satisfy specific luminance or chrominance criteria with an acceptable threshold. The area of ROI is given by Eq.5 where F represents the specific luminance or chrominance value, and M, N are the image dimensions in pixels.

$$Area_{ROI} = \sum_{ROI} F(m,n) \quad \text{…………………………..…………………….. (5)}$$

2.3. Visual Features of Flame Region in Digital Images

The following visual features are used in the literature to detect the flame region in an image [16], [34], [35]:

1) Brightness: the flame region stands in high brightness and contrast to its surroundings.
2) Spectrum: the flame region exhibits a structure of nested rings of colours, changing from white at the high energy core to yellow, orange, and red in the low energy periphery.
3) Motion: the uncontrolled fire region has a relatively static base (the burning body) at the bottom and smaller flames that keep flickering randomly on the top of the burning body.
4) The wavelet low frequency components of the fire region are relatively steady over time.
5) The wavelet higher frequency components of the fire region change in a stochastic fashion.
6) Temporal variations (fire flickering): Turbulent flames of an uncontrolled fire flicker with a frequency of 10 Hz.

7) Boundary shape: uncontrolled fire region should have a non-convex boundary.
8) Growth: the size of an uncontrolled fire region is evolving.

That was just a listing of the visual features of the flame region. Details on how these features are used in the fire detection algorithms will follow in the literature review sections within this chapter.

2.4. Background on Machine learning and Deep Learning

In the context of image processing, Machine Learning (ML) algorithms generally follow a standard flow, as shown in

Figure 2.3. A large number of images (Image Dataset) are used to study the region of interest (ROI). Extract and quantify the salient features of that ROI (image processing). The extracted features are stacked in the features array. Then, the Dataset is split into training subset (around 80% of the original dataset) used to train the model on the obtained features array and testing dataset (the remaining 20%) used to evaluate the model performance. Once the model is trained, evaluated, and exhibits acceptable performance metrics, it can be used on new images to detect, segment, or classify the same ROI.

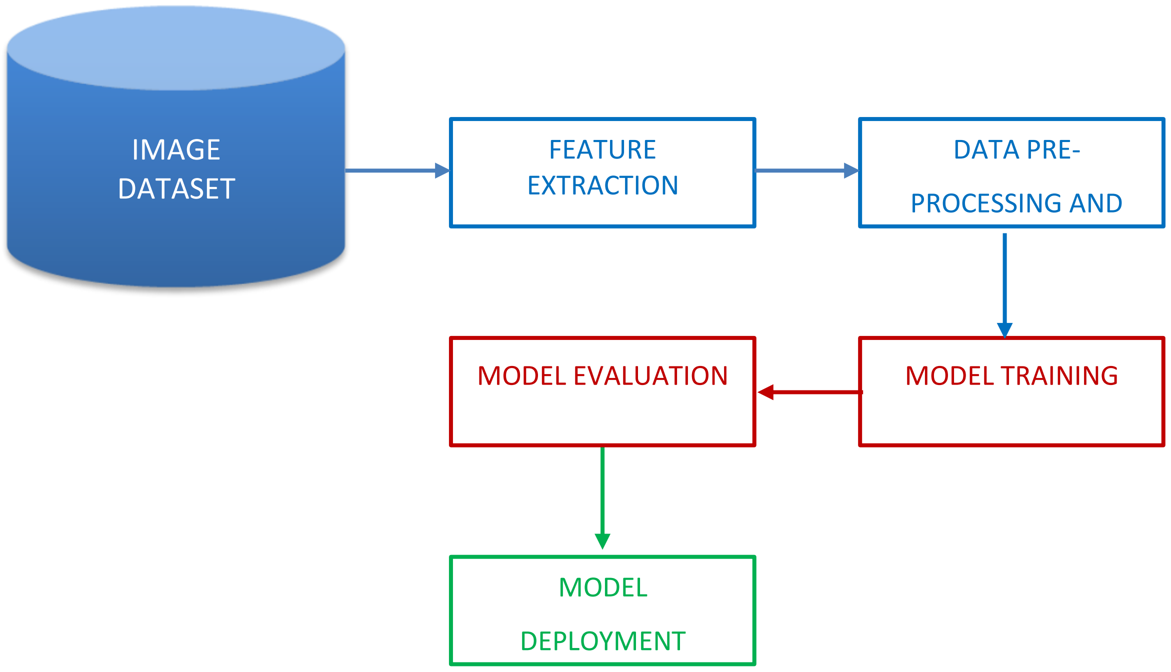


Figure 2.3 Machine Learning Flow Diagram.

2.4.1. Feature Engineering

The process of extracting and quantifying the salient features of the region of interest from images and saving them in the features array is known as feature engineering. The features array is then passed to the ML algorithm for object detection, segmentation, or classification.

2.4.2. Support Vector Machine Classification (SVM)

SVM is one of the most common traditional machine learning algorithms used in the literature. SVM is a mature supervised learning classification model, it aims to create a suitable hyper-plane (in multi dimensional space) to divide a given dataset into two parts with maximum margin of separation [27]. The maximum margin is essential so that the trained model can successfully classify, with a high degree of confidence, even the challenging new observations that lie at the margin between the two classes, Figure 2.4.

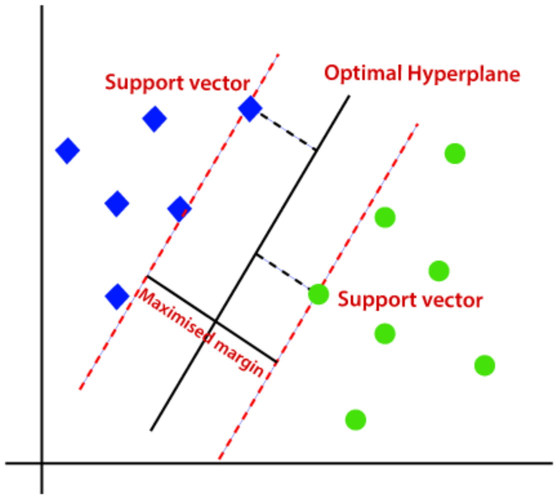


Figure 2.4 Visual representation for Support Vector Machine (SVM) classification in 2D.

Various kernel functions are used to implement the SVM model, e.g. linear, non-linear, polynomial, and radial basis function (RBF). The RBF is commonly used in image processing problems because it works the best for high-dimensional overlapping data problems like image classification. Furthermore, the RBF requires fewer parameters and saves training time [9]. RBF kernel is defined in Eq. 6, as follows:

$$k(x,y) = e^{\frac{-||x-y||^2}{2\delta^2}} \quad \text{...} \quad (6)$$

Where (x, y) are two different input feature arrays, and δ is a user defined parameter that scales the influence between the two input arrays. Smaller δ will lead to an overfit model which in turn will lead to misclassification for unseen data. On the other hand, higher δ will lead to a loose model which will also lead to misclassification. The best value of δ should be carefully chosen for an acceptable generalization error.

2.4.3. Deep Learning Image Classification

Deep learning (DL) is a modern subset of ML where a large amount of 2-dimensional (2D) labeled images (or the training dataset in general) are subjected to the multiple filters at each layer of the neural network. The 2D filter responses represent the features that are flattened to 1D feature vector at the last layer before passing to a binary classification layer that produces the binary output: Fire / Not-Fire. The process will be discussed in further detail in the next chapter.

2.4.4. The Difference Between DL and ML

The difference between deep learning (DL) and Machine learning (ML) is that feature extraction is done automatically and inherently within the DL classification model. While in ML the feature extraction is a process done separately before applying the classification model, Figure 2.5.

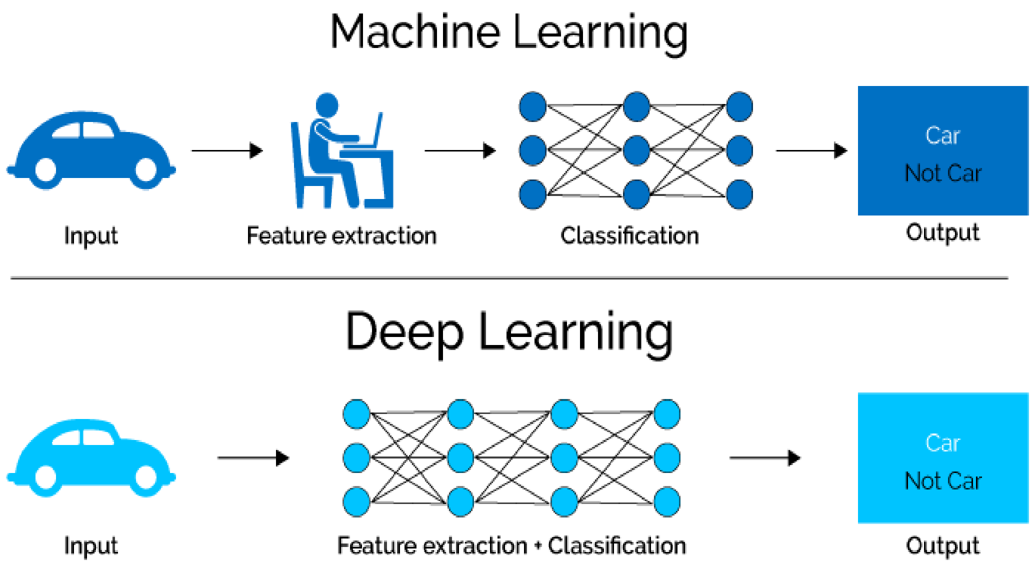


Figure 2.5 ML vs DL.

Recently, DL has gained substantial motive because of the humongous and exponential progress in computation hardware in terms of data processing and data storage capabilities, in addition to developments in data transmission (G5 networks) and the significant increase in utilization of the IoT sensors, with a reduction in the cost. Along with advances in the design of network structures and training methods.

DL has been in the heart of the implementation of modern fields such as natural language processing (NLP), medical diagnosis, self-driving cars, and computer vision, to name a few. Some DL applications have been outperformed human experts on some occasions [15], [32].

In the image classification applications, CNN has outpaced and showed superior performance compared with traditional approaches because of its powerful feature extraction techniques and robust model structure. Consequently, and gradually, traditional computer vision methods are being replaced by deep learning methods in the most recent research studies.

2.4.5. The Confusion Matrix

The concept of confusion matrix is related to the binary image classification problems such as fire or non-fire classification. In such case, there are only four outcomes, [4], [11], and [36]:

a) True-Positive (TP) when true fire condition correctly classified as fire;
b) True-Negative (TN) when true non-fire condition correctly classified as non-fire;
c) False-Positive (FP) when true non-fire condition misclassified as fire; and
d) False-Negative (FN) when true fire condition misclassified as non-fire.

The values of the true-positive (TP), true-negative (TN), false-positive (FP), and false-negative (FN) are better represented by the Confusion Matrix as shown below:

	Ground truth: Real Fire	**Ground truth: Not Fire**
Detected as Fire	TP	FP
Detected as non-Fire	FN	TN

2.4.6. Performance Metrics of Binary Classification Models

Evaluating the performance of a classification/prediction model can be tricky. Several concepts and measures are made to reflect the probability that the output of that model is informed versus chance (i.e., output obtained randomly). Using a single tool to measure the model performance can be misleading in case the dataset used to train and validate that model is biased towards one class over the other. Therefore, the performance metrics should be used with a clear understanding of their meanings. Commonly used model-performance metrics include Accuracy, Recall, Precision, and F-Score. The equation to calculate each metric and its meaning in the context of fire/non-fire image classification problem are given below:

Accuracy:

Accuracy [9] is the proportion of the correct classifications among the total number of classifications. Model Accuracy is given by Eq.7 as follows:

$$Acc = \frac{TP+TN}{TP+FP+TN+FN} \quad \ldots\ldots (7)$$

Precision, Recall, and F-Score:

While the Accuracy metric measures the overall performance of the classification/prediction model, the Precision, and the Recall metrics [22] focus on the relevancy of the output of the model. Precision is the proportion of the true fire detected by the model to the total fire (true or false) detected by the model. Model Precision is given by Eq. 8, as follows:

$$Precision = \frac{TP}{TP+FP} \quad \ldots\ldots (8)$$

Recall (also called Sensitivity) is the proportion of the true fire detected by the model to the total number of fire instances in the dataset (detected or not). Model Recall is given by Eq. 9, as follows:

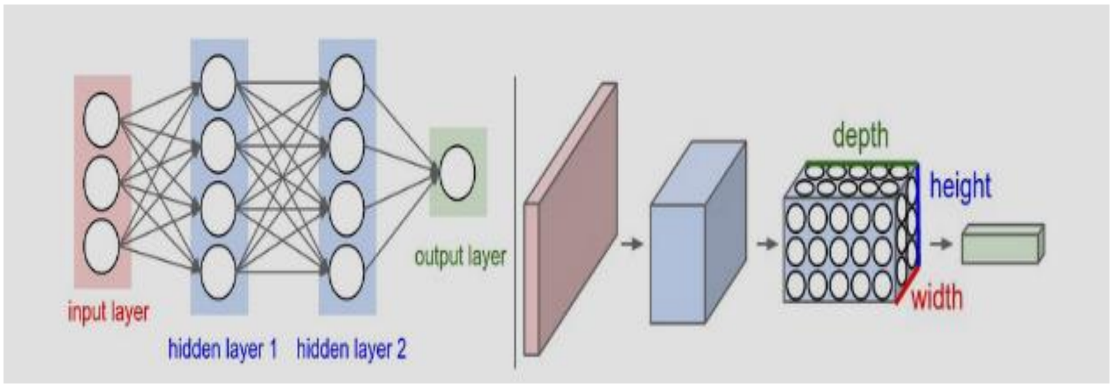
$$Recall = \frac{TP}{TP+FN} \ldots\ldots (9)$$

F- Score (F- Factor) [22] is another performance metric that is not directly related to the model output. Instead, it is related to both Precision and Recall. F- Score is given by Eq. 10, as follows:

$$F\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (10)$$

2.4.7. ANN and CNN

The name and structure are inspired by the human brain's neural cells. The artificial neural network (ANN) model is a series of algorithms (layers) that endeavors to recognize the underlying relationships (features) in an input (labelled) dataset through a process that mimics the technique the human brain operates. The simplest computational operations are performed in smaller units called neurons (aka nodes). One layer is formed of a group of neurons connected to other neurons in the former and successor layers, Figure 2.6a. Convolutional Neural Networks (CNN) is a special type of ANN that is designed to process 2D input data (i.e. images), Figure 2.6b.



(a) Generic ANN structure (b) 3-layer CNN structure

Figure 2.6 CNN is a special case of ANN [30].

2.4.8. Forward and Backward Propagation Paths of the ANN

The basic idea behind the artificial neural network (ANN) classification model is that each input in the input array contributes to the value of the ultimate output array via a group of weights and biases assigned to the relations with the neurons of the middle (hidden) layers. The middle layers calculate intermediate features. These features are then fed to the output layer (the feed-forward path) for the final classification. All neurons from one layer are connected to all neurons in the next layer to achieve the contribution.

The ANN is a supervised learning algorithm that manipulates labelled data. The input array and the corresponding known output (the desired output) are used to train the ANN model to determine the best fit values of weights and biases to minimize the difference between the calculated output and the desired output. That difference is known as the error function (or cost function). Several methods are used to calculate the error function. The most common method is the sum of the squared differences between the individual desired output and the corresponding calculated output. Backpropagation is the algorithm to determine the effect of a single training example to push the weights and biases, not just in terms of whether they should go up or down, but in terms of what relative proportions to those changes cause the most rapid decrease to the error (gradient descent).

The calculated output is obtained through the computations of the complete forward propagation path. Then the error is calculated and backward propagated from the final output layer to the input layer. The weights and biases are updated during the backward path according to the gradient descent rule. Then the forward path is repeated, and the new error is calculated, and so on. The process continues for a specific number of iterations or till the model converges, and the error becomes lesser than a predetermined threshold. Then, the best-fit weight values are determined. By then, the testing and validation phase begins to test the generalization performance of the model. Finally, the trained ANN model is used to classify new data.

2.4.9. The Components of The CNN Image Analysis Model

The main components that are used to build a CNN image analysis model are presented below. Further details on how those components are combined and optimized will be demonstrated in Chapters 3, 4.

2.4.9.1. Convolution Layer

The convolution (Conv) layer is the heart of a CNN model that does most of the heavy computations. It has three dimensions; the height and width are equal to the height and width of the input image (in pixels) and the depth is a number of trainable 2D spatial filters. Every filter (aka kernel) is a small window that has a width, height, and depth of 3 in the case of coloured images (since the coloured images have the depth of 3 colour channels). The kernel slides (convolute) over the full size of the input image. At each position, calculate the dot products between the entries of the kernel and the corresponding image pixels at that position (linear transformation), Figure 2.7. As the kernel scans the full width and height of the input image, another 2D activation map is created, that gives the responses of that filter at every spatial position. Single Conv layer usually includes many filters, not just one, typically 12, 24, 32, 64, 128 or even more filters depending on the complexity of the extracted features. Eventually, the network will learn the filters that activate when they see some type of visual features such as colour blobs, or an edge of some orientation, or even certain patterns on higher layers of the network [30]. The 2D output of the first Conv layer will be passed to the successive Convolution layers for more complex feature extraction. Usually, the CNN model involves multiple Convolution layers to extract more and more complex features and semantic object recognition.

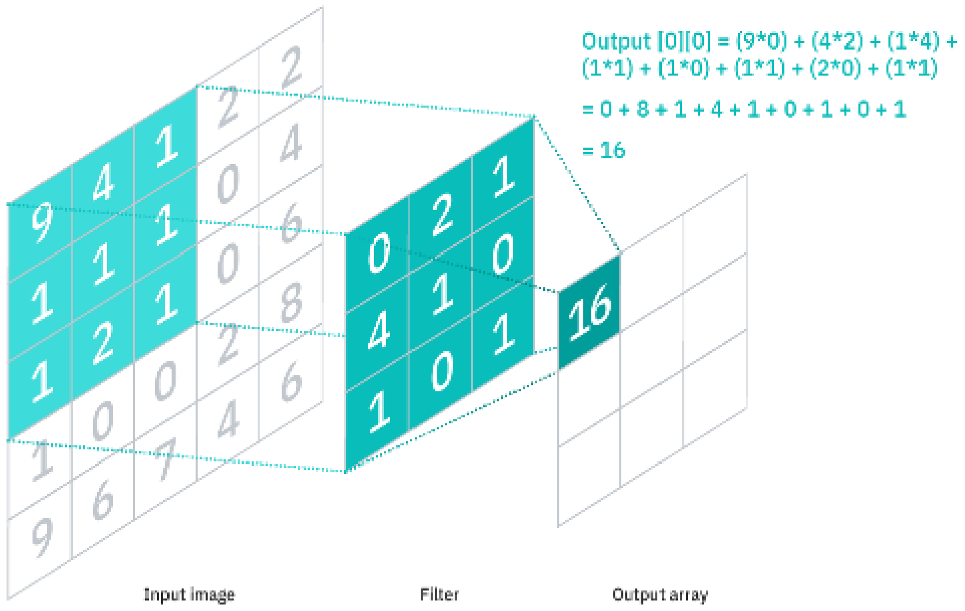


Figure 2.7 The Process of Convolution, visual presentation.

Dilated Conv is a special type of Conv where instead of increasing the size of the filter (kernel) to cover larger area of the input image, the same kernel is dilated leaving holes inside it, Figure 2.8. Dilated Conv provides a global view of the image but with the same number of parameters of the original small kernel, which improve the computation efficiency [41] especially when dealing with high resolution images where the nearby pixel values do not provide much information.

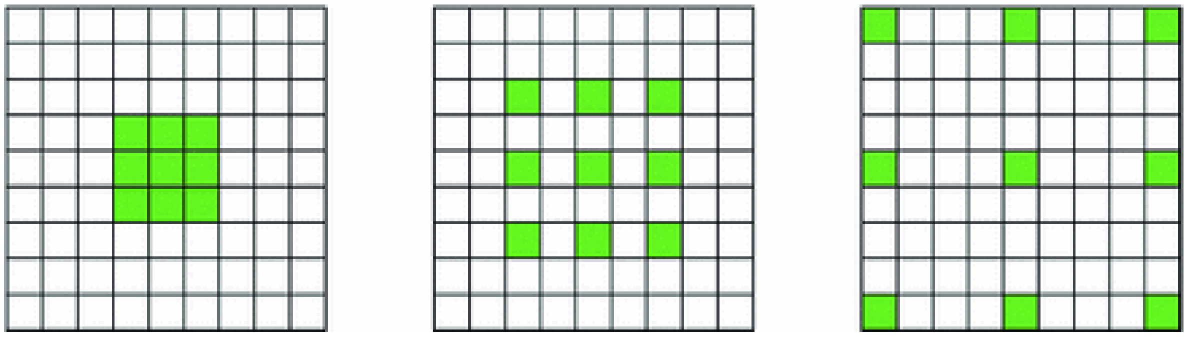


Figure 2.8 Dilated Convolution, visual presentation.

2.4.9.2. Rectified Linear Unit (ReLU) Activation Function

Each Conv layer is usually followed by a ReLU activation function to apply elementwise non-linearity, such that the relation between the input and output of single neuron is not limited to linear relationships. Relu activation function more fits the image processing applications as it doesn't produce negative output,

Figure 2.9. In case the output of the linear transformation is less than zero, the ReLU function will give zero, meaning that the output of this specific neuron will be ignored. The ReLU function is defined in Eq. 11, as follows:

$$y = \begin{cases} 0 & if\ x < 0 \\ x & if\ x > 0 \end{cases} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ldots\ldots (11)$$

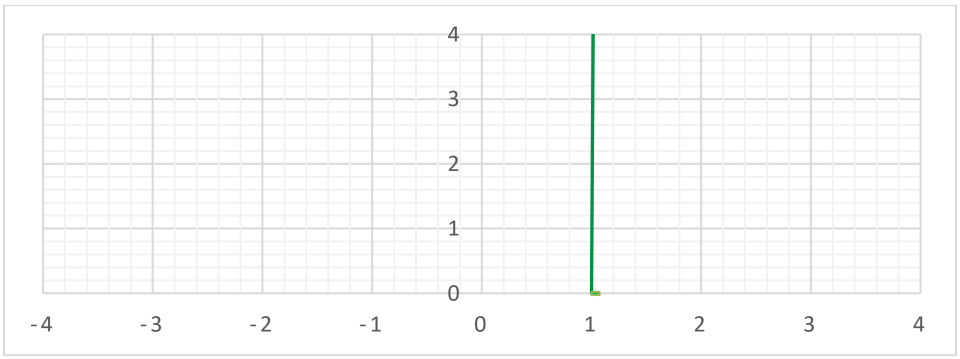


Figure 2.9 The ReLU activation function.

2.4.9.3. Pooling Layer

The pooling layer is inserted between Conv layers, and it replaces each batch in the input with a single output, which is the maximum of the input batch in case of max pooling (can also be average). A pooling layer effectively down samples the output of the prior layer, reducing the number of operations required for all the following layers especially when processing large size high definition and high-resolution images.

2.4.9.4. Normalization Layer

The normalization layer scales the input image pixels so that the output values remain between 0 and 1. The normalization process allows for faster calculations and improves the model stability and convergence.

2.4.9.5. Dropout Layer

One of the biggest problems with the CNN models is overfitting, which means the model performs perfectly fine and exhibits high accuracy only when processing the training data. Conversely, the model performs poorly when dealing with new data. Thankfully, the dropout layer has been introduced to solve this problem by deliberately dropping out the output of some neurons randomly during the model training phase. It does so by setting the inputs to those neurons to zero. The number of those dropped neurons is chosen by the model designer, as a fraction (between 0 and 1) of the total number of inputs to the layer. The dropout layer eliminates overfitting and improves the model generalizability.

2.4.9.6. Fully Connected/ Dense Layer

As mentioned earlier, the Conv layers deal with 2D images to extract their features in the form of 2D feature images. Those 2D feature images are Converted to 1D arrays through a process called flattening. Then the flattened feature vector is fully connected to the final layer (the classification layer). The purpose of the fully connected layer (aka dense layer) is to take into account each pixel value in the feature image. By doing so, the model accuracy is significantly improved, and the number of misclassifications is reduced.

2.4.9.7. Sigmoid Activation Function

The final step after the last fully connected layer is the activation function that decides the output of the overall model. There are two common activation functions used in the literature at the final layer, Sigmoid and SoftMax [30]. The sigmoid function is used in binary classification problems where the output looks like fire or non-fire, healthy or non-healthy, 1 or 0, etc., Figure 2.10

The sigmoid function is defined by Eq. 12, as follows:

$$y = \frac{1}{1+e^{x}} \quad \text{……………………………..…………..…………..…………..…………..…..} \quad (12)$$

The sigmoid function gives two probability values representing the degree of output belonging to each class. The sum of the two probability values must be equal to one. Then the final classification goes to the higher probability class. The SoftMax function works the same way, except it produces more than two probability values, and hence it is used in multi-class classification problems.

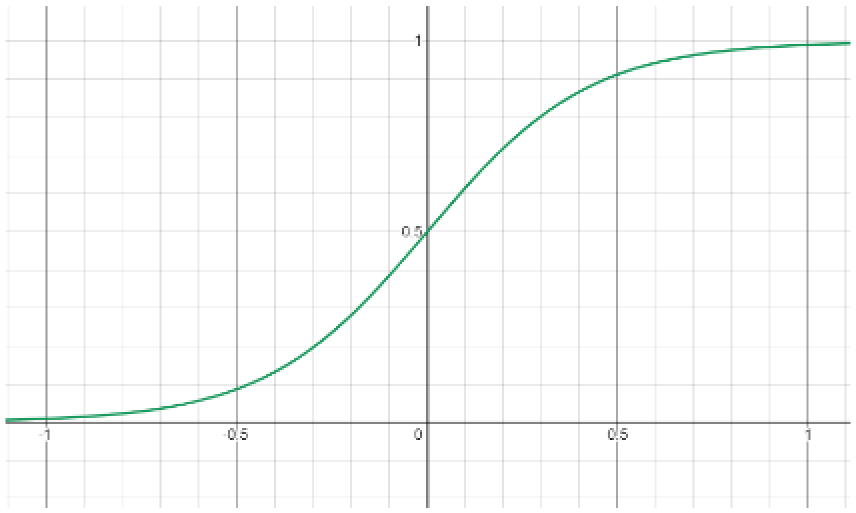


Figure 2.10 The Sigmoid activation function.

2.5. Recent Image Processing Algorithms for Fire Detection

The reviewed literature is classified into four main approaches:

1. Fire detection based on image processing (colour and brightness information only)
2. Fire detection based on video processing (colour and motion information)
3. Fire detection based on advanced video processing (colour, motion, and spatiotemporal frequency analysis)
4. Smoke detection

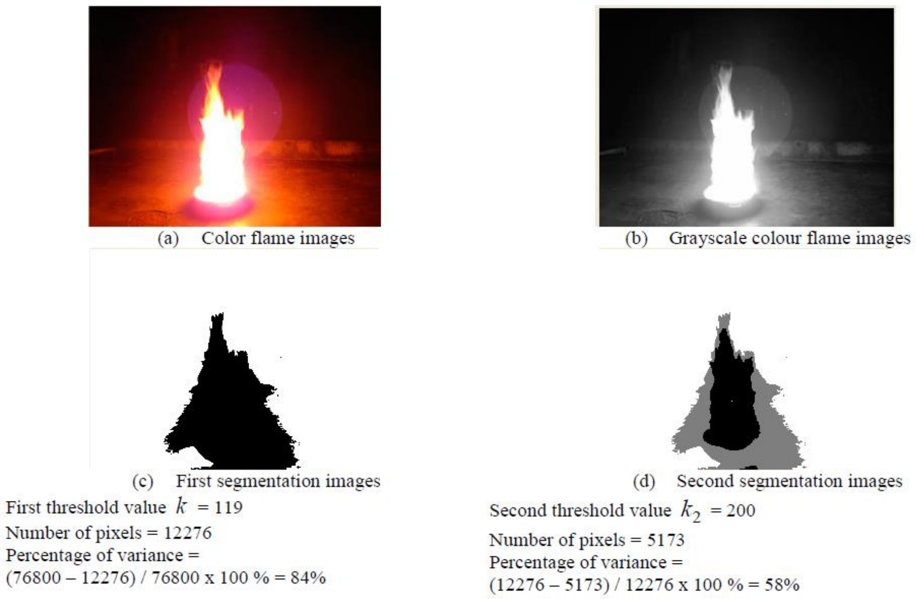
In the following sections, each approach will be explored. The different fire detection algorithms and techniques used in recent research will be reviewed and evaluated.

2.5.1. Fire Detection Based on Image Processing (Colour & Brightness Information)

In a recent research, Wong & Fong (2014) proposed multi-thresholding segmentation technique to divide the image into: a) Dark background and b) Bright blob represents the edges of fire coloured region. Since the hot core of the real fire region is always brighter than the edges, then the brightest blob represents the core of the fire-coloured region,

Figure 2.11.

.



(a) Color flame images
(b) Grayscale colour flame images
(c) First segmentation images
First threshold value k = 119
Number of pixels = 12276
Percentage of variance =
(76800 – 12276) / 76800 x 100 % = 84%
(d) Second segmentation images
Second threshold value k_2 = 200
Number of pixels = 5173
Percentage of variance =
(12276 – 5173) / 12276 x 100 % = 58%

Figure 2.11 Multi-thresholding segmentation for the fire region.

The centroid for this bright blob is calculated, manually labelled as real fire or fire-like object and saved in a database. Next, the nearest neighbour classification algorithm is trained on the labelled database and then deployed to predict whether the new image represents a flame image or a flame-like object image. The major drawback of this algorithm is it requires a lot of training to achieve reliable results. In addition, the centroid displacement is not enough to differentiate between a real fire and a flame-coloured object that slowly oscillates in front of the camera.

Vipin (2012) proposed a seven rules colour-based technique to detect forest fire in RGB and YCbCr colour spaces, that is for a pixel in an image to be classified as fire pixel the following conditions must be all satisfied:

1. In RGB colour space, the value of the red channel (R) is greater than the value of the green channel (G) which in turn is greater than the value of the blue channel (B). This is expressed in Eq 13, as follows:

$$R(m,n) > G(m,n) > B(m,n) \quad \text{……………………………..…….……… (13)}$$

2. The values of the red and the green channels (R, G) are greater than prespecified threshold values (Th1, Th2) and the value of the blue channel (B) is less than a third threshold value (Th3). This is expressed in Eq 14, as follows:

$$R(m,n) > Th1 \cap G(m,n) > Th2 \cap B(m,n) < Th3 \quad \text{……………………..… (14)}$$

3. In YCbCr colour space, the value of the luminance component (Y) is greater than the value of the chrominance blue component (Cb). This is expressed in Eq 15, as follows:

$$Y(m,n) > Cb(m,n) \quad \text{……………………………………………………..…..….… (15)}$$

4. The value of the chrominance red component (Cr) is greater than the value of the chrominance blue component. This is expressed in Eq 16, as follows:

$$Cr(m,n) > Cb(m,n) \quad \text{………………………………………...….…………….. (16)}$$

5. The values of luminance and chrominance red components are greater than their mean values (Y_{mean} and Cr_{mean}) respectively in the overall image and the value of chrominance blue component is less than its mean value (Cb_{mean}). This is expressed in Eq 17, as follows:

$$Y(m,n) > Y_{mean} \cap Cr(m,n) > Cr_{mean} \cap Cb(m,n) < Cb_{mean} \quad \text{……..…. (17)}$$

6. The absolute difference between the values of the chrominance red and the chrominance blue components is greater than or equal to a prespecified threshold. This is expressed in Eq 18, as follows:

$$|Cr - Cb| \geq Th \quad \text{……} (18)$$

7. The values of chrominance red and the chrominance blue components are limited by threshold values obtained empirically. This is expressed in Eq 19, as follows:

$$[Cr(m,n) \geq 150] \cap [Cb(m,n) \leq 120] \quad \text{……} (19)$$

The model produces acceptable results in fire detection mainly because it uses the YCbCr colour space, which separates luminance and the chrominance components, which in turn makes the model more robust to changes in illumination conditions. However, knowing the linear relation between the RGB and the YCbCr colour spaces, some of the mentioned rules are redundant, for example, if rule 1 is true, then rule 3 is automatically true as well.

Sharma et al (2020) reused Vipin's rules after removing the redundancies. So the colour segmentation rules are reduced to three. These are expressed in Eq. 20, Eq 21, and Eq 22, as follows:

Rule 1: $Y(m,n) > Cb(m,n) \;\cap\; Cr(m,n) > Cb(m,n)$ …… (20)

Rule 2: $Y(m,n) > Y_{mean} \;\cap\; Cr(m,n) > Cr_{mean} \;\cap\; Cb(m,n) < Cb_{mean}$ …… (21)

Rule 3: $|Cr(m,n) - Cb(m,n)| \geq Th$ …… (22)

Premal & Vinsley (2015) took a different approach while using very similar colour segmentation rules in the YCbCr colour space. Those are expressed in Eq. 23, Eq 24, Eq 25, and Eq 26, as follows:

Rule 1: $Y(m,n) > Cb(m,n) \cap Cr(m,n) > Cb(m,n)$ …… (23)

Rule 2: $Y(m,n) > Y_{mean} \cap Cr(m,n) > Cr_{mean} \cap Cb(m,n) < Cb_{mean}$ …… (24)

Rule 3: $Cb(m,n) \geq Y(m,n) > Cr(m,n)$ …… (25)

Rule 4: $Cr(m,n) \leq (\tau \times Cr_{std.dev})$ …… (26)

Where $Cr_{std.dev}$ is the standard deviation for the chrominance red component and value of τ is a constant determined empirically by studying manually segmented fire region in sample fire images. The fire detection algorithm works as follows:

1. If the examined pixel satisfies both rule 1 and rule 2, then it may represent the hot edge of fire region, and
2. If the examined pixel satisfies both rule 3 and rule 4, then it may represent the very hot white core of the fire region.

Finally, if both conditions 1 and 2 found to be true in one image, then a true fire region exists in that image.

Figure 2.12 a, b, c show the input RGB image and the fire region segmentation based on conditions 1 and 2. The final fire region combining both conditions (1, 2) is shown in

3. Figure 2.12 d

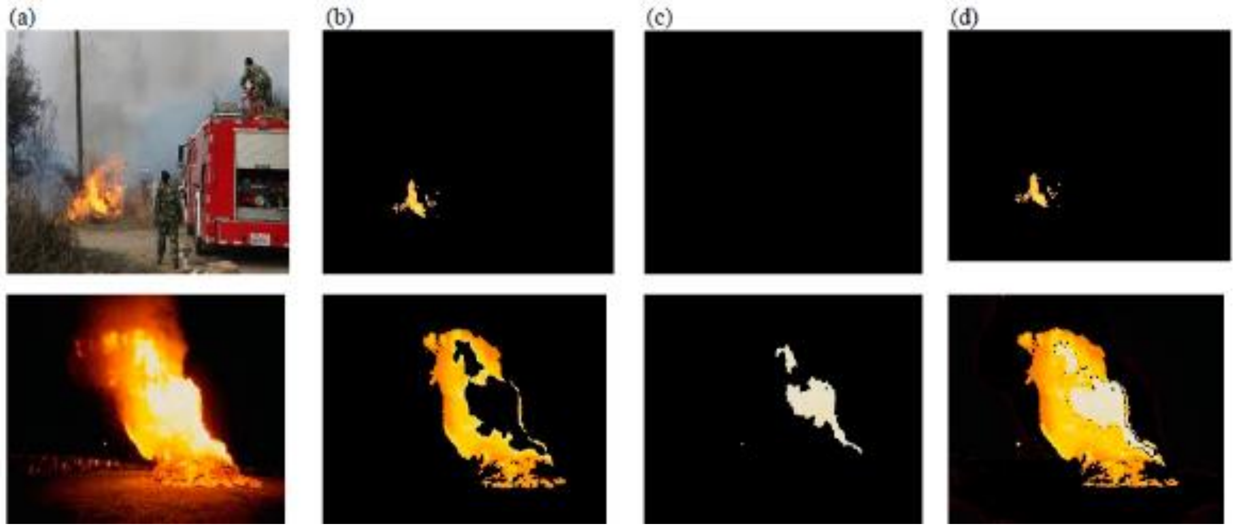


Figure 2.12 Fire Region Segmentation [24].

This approach is very sensitive to the environmental and illumination conditions. Meaning that, the results of the colour segmentation processes are severely affected by the time of the fire incident, day versus night, and the overall average brightness of the image. Furthermore, the approach cannot distinguish between a real fire and non-fire bright objects such as street lighting. Therefore, this approach is not reliable for fire detection in urban communities.

Çelik & Demirel (2009) presented a similar approach based on colour rules in YCbCr colour space. The rules are expressed in Eq. 27, Eq. 28, Eq. 29, and Eq.30, as follows:

Rule 1: $Y(m,n) > Cb(m,n)$ ……..……………………………………..………..….. (27)

Rule 2: $Cr(m,n) > Cb(m,n)$ ……..……………………………….………..…..… (28)

Rule 3: $Y(m,n) > Y_{mean} \cap Cr(m,n) > Cr_{mean} \cap Cb(m,n) < Cb_{mean}$ ……………… (29)

Rule 4: $|Cr(m,n) - Cb(m,n)| \geq \tau$ ……..……………………………………..…..… (30)

Where, τ is a constant determined empirically by studying sample fire images with manually segmented fire regions. The value of $\tau = 40$ is chosen, at this value the detection rate is over 90% and false alarm rate is less than 40%.

Then, by analyzing manually segmented fire regions in 1000 images; and using the least-square estimation technique, the authors derived three polynomials that bound the fire pixels. Now, for a given pixel to be classified as a fire pixel, its values must satisfy the above four colour-based rules and lie in the region bounded by the three polynomials. The authors claimed that their proposed colour model achieves a 99.0% flame detection rate and a 31.5% false alarm rate. However, the derivative polynomials are biased by the training images, which means the system will exhibit a high probability of false alarms in case of new photos classification.

In general, colour-based segmentation alone may give modest results in forest fire detection applications. However, this technique is totally unreliable to detect fire in buildings or in urban areas such as city streets, roadways, construction sites, parking lots, workshops, hangars, etc. where there are a large number of fire-coloured objects moving around, which will lead to countless number of false alarms. Examples of fire-coloured moving objects are people with fire-coloured clothes, fire-coloured vehicles, vehicle headlights, street lights, to name a few. Furthermore, those technique deals only with still images and miss the extensive information implicit in the sequence of frames in a video stream.

2.5.2. Fire detection Based on Video Processing (Colour and Motion/Growth Information)

Seebamrungsat et al. (2014) proposed a colour-based segmentation algorithm by empirically estimating colour thresholding values in HSV colour space. Rules for segmentation of the fire-coloured pixels are expressed in Eq. 31, thru Eq. 36, as follows:

1. In HSV colour space, the fire-coloured pixel must satisfy the following rules:

   a. the value of luminance component (V) lies in this range:

   $0.98 < V(m,n) < 1.00$ …………………………………………………………… (31)

   b. the value of saturation component (S) lies in this range:

   $0.20 < S(m,n) < 1.00$ …………………………………………………………….. (32)

   c. the value of hue component (H) lies in this range:

   $0.02 < H(m,n) < 0.30$ …………………………………………………………….. (33)

2. in YCbCr colour space, the fire-coloured pixel must satisfy the following rules:

   a. the value of chrominance red component (Cr) is greater than or equal to its mean value (Cr_{mean}) in the overall frame; $Cr(m,n) \geq Cr_{mean}$ …………………………….. (34)

   b. the value of chrominance blue component (Cb) is less than or equal to its mean value (Cb_{mean}) in the overall frame; $Cb(m,n) \leq Cb_{mean}$ …………………………………….. (35)

   c. the value of luminance component (Y) is greater than or equal to its mean value (Y_{mean}) in the overall frame; $Y(m,n) \geq Y_{mean}$ ………………………………………..…..….. (36)

In order to distinguish between a real fire and a fire-coloured objects, the growth of the segmented fire coloured region is compared between five successive frames, if the region is growing, then it represents a real fire spreading in the scene. Else, it is non-fire objects such as people with red/orange clothes, yellow-coloured vehicle, etc.

This approach is not robust as it depends mainly on colour threshold values to extract the flame-like objects. The thresholding values are challenging to estimate and widely vary with the illumination and weather conditions. In addition, the growth checking is not efficient since fire-coloured objects moving towards the camera will look like growing.

Wang et al. (2013) took one step beyond the colour and growth examination by checking the randomness motion of the flame region as extracted from a sequence of frames in a video stream. First, in the offline training phase, the authors estimated the probability of flame colour values using Gaussian distribution modelled in the YCbCr colour space. This technique has an advantage over the previously mentioned colour segmentation rules, as it eliminates the need for pre-set threshold values to detect the fire-coloured region.

Second, the foreground moving object is detected in real time by dynamically estimating the background using the approximate median algorithm. In this algorithm, the temporal median of each pixel's brightness is calculated in grayscale for the past n-frames. All pixels' median values represent the temporary estimated background. Then, the foreground moving object is obtained by subtracting the current frame from the estimated background. So, if the pixel value in the current frame is far from its median (greater than a threshold value), then this pixel is classified as a foreground pixel. In other words, it is a moving pixel. Otherwise, if the current pixel value is close enough to its temporal median, that pixel is classified as a background pixel. The mathematical representation of moving region extraction is given in Eq. 37 below, consider F (m, n, t) is the current frame, and F (m, n, t-i) for $i \in 1,2,3, \ldots, n$ are the previous n-frames, all are in grayscale, then: The Moving $\text{region} = |F(m,n,t) - Median\ of\ \{F(m,n,t-i)\}|$ ………………… (37)

Third, the candidate flame region is obtained by intersecting the segmented colour region and the moving region. This interpreted mathematically as applying the logic AND operator to both regions obtained by colour segmentation and motion segmentation. The AND operation is expressed in Eq. 38, as follows:

$\text{Candidate } Flame\ Region\ = Color\ Segmetation \text{ region} \cap Moving \text{ region}$……....…. (38)

Next, for each flame candidate, consider, F_c is the grayscale brightness of that candidate, and (m, n) are its spatial coordinates. The following seven features are calculated to form the feature vector:

1. The mean value of the Gaussian distribution of chrominance-red component (μ_{Cr});
2. The mean value of the Gaussian distribution of chrominance-blue component (μ_{Cb});
3. The standard deviation of the chrominance-red component (σ_{Cr});
4. The standard deviation of the chrominance blue component (σ_{Cb});

5. Area of the Candidate Flame Region, this is given in Eq. 39, as follows:

$$A_t = \sum_{m,n} F_c(m,n) \text{ ………………………………………………………………… } (39)$$

6. The X-coordinate of the centroid of the Candidate, this is given in Eq. 40, as follows:

$$X_t = \frac{1}{A_t} \sum_{m,n} (m * F_c(m,n)) \text{ ……………………………………………………… } (40)$$

7. The Y-coordinate of the centroid of the Candidate, this is given in Eq. 41, as follows:

$$Y_t = \frac{1}{A_t} \sum_{m,n} (n * F_c(m,n)) \text{ ……………………………… ……………..……… } (41)$$

In the final step, Wald–Wolfowitz randomness test is applied on each feature vector to confirm whether it is representing a real fire, or it is just a fire-coloured moving object. This technique shows its adaptability to various environmental and illumination conditions; however, it cannot detect bright flame on dark background such as fire at night [38].

Zhang et al (2007) suggested a new algorithm that deals with videos taken by a moving or fixed camera. The authors defined four frame features to test the similarity between frames: colour histogram, dominant colours, percentage, and average of dominant colours. Let I_k^{iR} represents k-dimension of red colour histogram in frame i. The similarity features are given by Eq. 42 thru Eq. 45. If the similarity between two frames falls within a predefined threshold, they are considered similar frames.

1. Similarity between two frames (i, j) based on Colour Histogram feature in RGB space is calculated as:

$$\text{Sim}_{CH}(i,j) = \frac{\sum_{k=1}^{256} min\left(I_k^{iR}, J_k^{jR}\right) + \sum_{k=1}^{256} min\left(I_k^{iG}, J_k^{jG}\right) + \sum_{k=1}^{256} min\left(I_k^{iB}, J_k^{jB}\right)}{\sum_{k=1}^{256} I_k^{iR} + \sum_{k=1}^{256} I_k^{iG} + \sum_{k=1}^{256} I_k^{iB}} \text{ ……………..….. } (42)$$

2. Similarity between two frames (i, j) based on Dominant Colour feature in RGB space is calculated as:

$$\text{Sim}_{DC}(i,j) = \frac{|R\_DC^i - R_D C^j| + |G\_DC^i - G_D C^j| + |B\_DC^i - B_D C^j|}{R_D C^i + G_D C^i + B_D C^i} \text{ ………………….…….. } (43)$$

3. Similarity between two frames (i, j) based on Dominant Colour Percentage feature in RGB space is calculated as:

$$\mathrm{Sim}_{DCP}(i,j) = \frac{|R\_DCP^{i} - R_{D}CP^{j}| + |G\_DCP^{i} - G_{D}CP^{j}| + |B\_DCP^{i} - B_{D}CP^{j}|}{R_{D}CP^{i} + G_{D}CP^{i} + B_{D}CP^{i}} \quad \text{.................. (44)}$$

4. Similarity between frames based on Dominant Colour Average feature in RGB space is calculated as:

$$\mathrm{Sim}_{DCA}(i,j) = \frac{|R\_DCA^{i} - R_{D}CA^{j}| + |G\_DCA^{i} - G_{D}CA^{j}| + |B\_DCA^{i} - B_{D}CA^{j}|}{R_{D}CA^{i} + G_{D}CA^{i} + B_{D}CA^{i}} \quad \text{.................. (45)}$$

Finally, the total similarity, $Sim(i,j)$, between two frames, (i,j), is calculated as the weighted sum of the above similarity values. Let $W_{CH}, W_{DC}, W_{DCP}, and\ W_{DCA}$ are the weight coefficients, their sum is accumulated to one. The total similarity is given by Eq. 46, as follows:

$$Sim(i,j) = W_{CH} \times \mathrm{Sim}_{CH}(i,j) + W_{DC} \times \mathrm{Sim}_{DC}(i,j) + W_{DCP} \times \mathrm{Sim}_{DCP}(i,j) + W_{DCA} \times \mathrm{Sim}_{DCA}(i,j) \quad \text{.. (46)}$$

The first frame of the sequence of similar frames is the key frame which represents that frame set. As a result, key frames are extracted to index the video division, and the following colour rules are applied on the key frames to detect the occurrence of the fire.

Next, flame colour rules are defined in the HSV colour space. Intuitively, it is noticeable that the colour of high energy fire is bright white, whereas the low energy fire produces saturated colour somewhere between red and yellow, Figure 2.13. The colour rules in HSV colour space are given by Eq. 47 thru Eq. 50, as follows:

Rule 1, range of the Hue component between $H_{Low} = 0°$ (red) and $H_{High} = 40°$ (yellow):

$$0 < H(m,n) < H_{High} \quad \text{.. (47)}$$

Rule 2, range of the Saturation component between $S_{Low} = 0.1$ and $S_{High} = 0.3$;

$$S_{Low} < S(m,n) < S_{High} \quad \text{.. .... (48)}$$

Rule 3, range of the Value component (the brightness) between $V_{Low} = 0.5$ and $V_{High} = 0.8$:

$$V_{Low} < V(m,n) < V_{High} \quad \text{.. (49)}$$

$$\text{Eventually} \quad \begin{cases} if \quad Rule\ 1\ \cap\ Rule\ 2\ \cap\ Rule\ 3 \quad Then\ this\ is\ a\ flame\ pixel \\ Else,\ this\ is\ non\ flame\ pixel \end{cases} \quad \text{...........(50)}$$

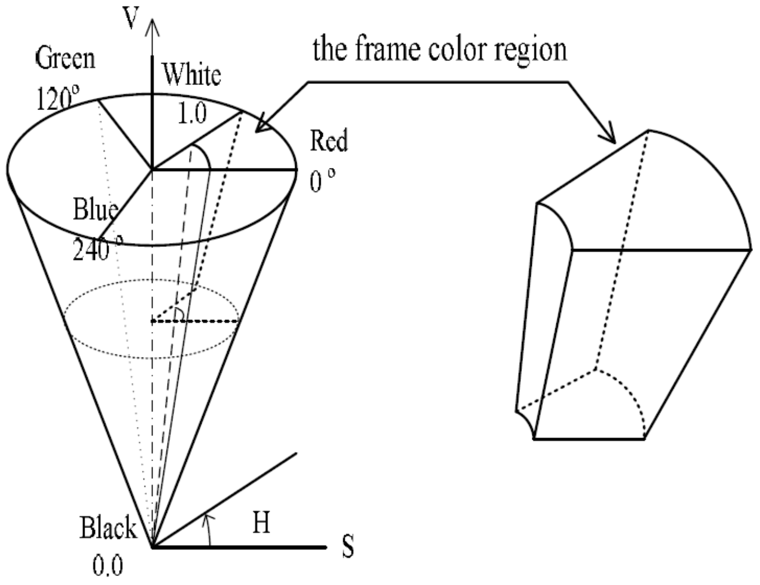


Figure 2.13 Visual representation of flame colour region in HSV space. [44]

The last rule, Rule 4, threshold the size of the flame area: $N > N_{min}$ where N_{min} is the minimum flame size to trigger the alarm, setting the threshold value helps to avoid false alarms caused by noise or outliers. The threshold values are set empirically through analyzing many fire video-clips. The fire region is traced in each key frame to classify the frame content as fire or non-fire. Similar frames then clustered into one scene. Fire features are utilized to locate the starting frame and ending frame with fire.

This technique has one advantage that it does not require the recording camera to be stationary. It can deal with video captured by a moving camera. However, it suffers from three main disadvantages:

1) The expensive computation to extract the keyframes makes it not suitable for real-time fire detection application. It fits more the analysis of pre-recorded videos;
2) It cannot distinguish between a real fire and fire-coloured objects like red flags, yellow signs, etc.; and
3) The manually adjusted threshold values make the results highly sensitive to different illumination and environmental conditions. This high sensitivity to the different conditions causes false alarms and reduces the system accuracy and reliability.

2.5.3. Fire Detection Based on Advanced Video Processing (Colour, Motion, and Spatiotemporal Frequency Analysis)

It is useful here to refer to two frequency properties used in digital image and video processing: temporal frequency and spatial frequency. The temporal frequency refers to the variations of the values of single pixel over sequence of frames, whereas the spatial frequency refers to the variations of the values of different pixels in a single frame (image)

Liu & Ahuja (2004) proposed an algorithm to detect fire based on brightness, colour, spatial, and temporal variations according to the following steps:

First: the potential fire regions are extracted from an image using the fire spectral and spatial clues,

1. Extract the high brightness regions in grayscale frame (the seed regions), those possibly representing fire cores.
2. The extracted seeds are allowed to be stretched by following spectral gradients of the image and adding neighbor pixels if their colour values in HSV colour space lies within the fire colour model with sufficiently high probability. The fire colour probability density functions are modelled as a mixture of Gaussian distributions, Figure 2.14, in HSV colour space. The distributions (F) are estimated from past observations of various fire images. Consider $X = \begin{pmatrix} H(m,n) \\ S(m,n) \\ V(m,n) \end{pmatrix}$, $\mu = \begin{pmatrix} H_\mu \\ S_\mu \\ V_\mu \end{pmatrix}$, and $\sigma = \begin{pmatrix} H_\sigma \\ S_\sigma \\ V_\sigma \end{pmatrix}$ are HSV colour space values, their mean, and their standard deviation for the candidate pixel, respectively, then the distributions (F) is expressed in Eq.51, as follows:

$$F(X) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{X-\mu}{\sigma}\right)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(51)}$$

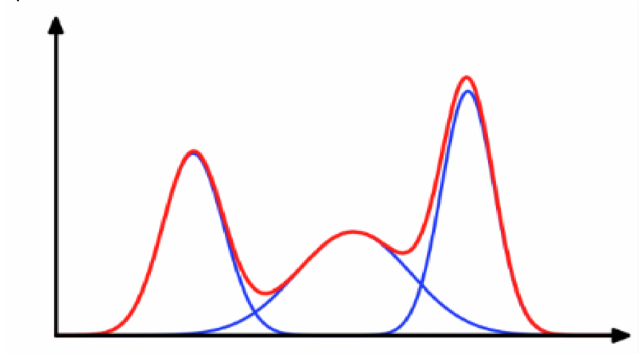


Figure 2.14 Mixture of three Gaussian distributions in 2D.

Second, the region boundaries are represented by Fourier transform coefficients (Fourier Descriptors). Fourier Descriptors represent the spatial frequency variations of each region. The coefficients of the Discrete Fourier Transform (DFT) of $\{z_i\}_{i=1}^{N}$ are give in Eq. 52, as follows:

$$a_k = \frac{1}{N} \sum_{i=1}^{N} z_i \cdot e^{-j\frac{2\pi}{N}ik} \qquad \text{for K= 0,1, 2, …, N-1} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots (52)$$

K is the number of coefficients or number of samples considered in the frequency domain. According to Nyquist sampling criterion, only half of the original samples are needed to perfectly describe the original signal/image. So, only M coefficients are required to represent the spatial frequency variations of the region boundary, where $M \leq -\left\lfloor \frac{N-1}{2} \right\rfloor$, the Floor value of half of the original sampling frequency.

Third, the temporal variation of each region is estimated using autoregressive model (AR) to forecast the future shape of each region as a function of its previous images in the image sequence. In its general form, the autoregressive model predicts the present value (F_t) of the candidate pixel based on its previous number of values in previous (p) frames, where (p) is an integer. This is expressed in Eq. 53, as follows:

$$F_t = C + \emptyset_1 F_{t-1} + \emptyset_2 F_{t-2} + \emptyset_3 F_{t-3} + \dots\dots + \emptyset_p F_{t-p} + \in_t \dots\dots\dots\dots\dots\dots\dots\dots\dots (53)$$

where C and $\in_t$ are the constant intercept and the error at the moment (t), respectively.

Finally, Fourier Descriptors and autoregressive model parameters are stacked in the features' vector that describes each region. Two-class Support Vector Machine (SVM) classification algorithm with radial basis function kernel is then used for fire region recognition.

According to the authors, this algorithm is able to distinguish between real fire regions and fire-coloured objects with a high degree of accuracy up to 99.9%. However, it cannot detect small fire regions or those fire regions far away from the camera [16].

Töreyin et al (2006) introduced a new approach that uses the shape variation and flickering as clues about the fire region beside the colour and the motion clues. Video captured by an ordinary fixed surveillance camera is processed in real-time as follows:

First rule, moving objects are detected by applying hybrid background estimation method on grayscale frames. The pixel at position (x, y) is classified as foreground or moving pixel if there is a significant change in its brightness between the current frame and the previous frame. Otherwise, the frame is considered to be background or stationery. This is expressed in Eq. 54, as follows:

$$\begin{cases} IF\ |F_t(m,n) - F_{t-1}(m,n)| > Th_t(m,n)\ THEN\ the\ pixel\ at\ position\ (m,n)\ is\ moving \\ ELSE,the\ pixel\ at\ position\ (m,n)\ is\ non-moving \end{cases} \quad \ldots\ldots (54)$$

Where $Th_t(m,n)$ is adaptive threshold. Then, the background at time (t) is estimated from the previous background using Eq.55, as follows:

$$B_t(x,y) = \begin{cases} \alpha B_{t-1}(m,n) + (1-\alpha)F(m,n)\ if\ the\ pixel\ (m,n)\ is\ moving \\ B_{t-1}(m,n)\ if\ the\ pixel\ (m,n)\ is\ non-moving \end{cases} \quad \ldots\ldots\ldots\ldots (55)$$

Where (α) is a learning factor close to one. Subtracting the background from the current frame gives the moving regions in that frame.

Second rule, the colours of the pixels in the current frame are compared to mixture of ten Gaussian distributions in [R, G, B] colour space, the ten Gaussian distributions are estimated from previous observations. If the pixel colour lies within two standard deviations from the center of Gaussians, then this pixel is classified as fire-coloured pixel.

The intersection of the moving regions obtained from the first step with the segmented fire-coloured regions obtained from the second step is called fire-coloured moving region(s) or fire candidates. Those fire candidates are presented to the third step as binary images (black and white).

Third rule, the temporal frequency of the resultant fire candidates is tested by applying temporal wavelet to detect the 10Hz flickering rate that characterizes the turbulent flame flickering [35]. Most standard cameras nowadays capture video with a rate of 25 or 30 frames per second, this is more than double of the flame flickering rate, hence aliasing will not happen. Therefore, the real-fire flickering rate can be observed at its natural frequency, the 10Hz. So, the temporal variations of one-dimensional signal of red colour component across the frames contain fire-coloured moving regions are analyzed using two-channel sub-band decomposition filter bank, Figure 2.15. The HPF and LPF represent half-band high-pass and low-pass filters with coefficients [-0.25, 0.5, -0.25] and [0.25, 0.5, 0.25] respectively [35]. The output high-band wavelet sub-signals

d_n (m, n) and e_n (m, n) represent the high frequency activity at pixel (m,n). If the sub-signals d_n (m, n) and e_n (m, n) have a few fluctuations (zero crossings per second) less than a pre-set threshold value, then they represent fire coloured moving pixel that slowly changes its value over time. On the other hand, if the sub-signals d_n (m, n) and e_n (m, n) have so many fluctuations (zero crossings per second) higher than the pre-set threshold value, then this is considered a strong clue about a real fire flickering at high frequency compared to a fire-coloured moving object.

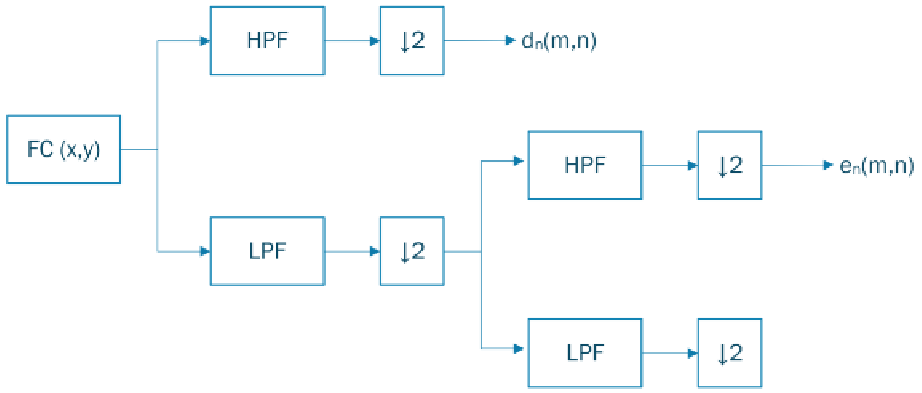


Figure 2.15 Wavelet two-stage filter bank.

Fourth rule, the spatial frequency is checked, the spatial wavelet analysis of a rectangular frame containing the pixels of fire-coloured moving regions is performed, Figure 2.16. By nature, the real fire region will exhibit significant spatial variations in colours, while the fire-coloured moving object will show much lesser spatial variations in colours.

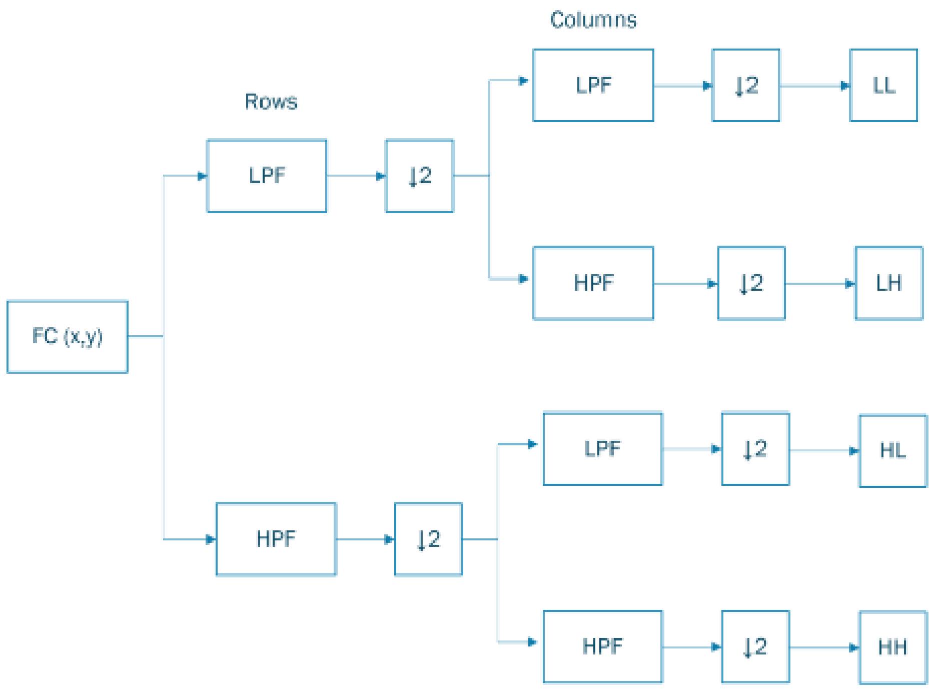


Figure 2.16 Spatial Wavelet Filter Bank Scheme – 1 Level Image Decomposition.

To quantify the spatial variation of colours, the total energy (E) of the wavelet details sub-images is calculated using Eq.56 below, where $M \times N$ is the size of the fire-coloured moving object, and x_{lh}, x_{hl}, x_{hh} are the wavelet details sub-images.

$$E = \frac{1}{M \times N} \sum_{x,y} |x_{lh}(m,n)|^2 + |x_{hl}(m,n)|^2 + |x_{hh}(m,n)|^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots (56)$$

So, the summary of the previous four rules is given below. The final decision is a linear weighted combination of the four rules (decision fusion) as expressed in Eq. 57:

R1 = 1 if the pixel is moving, and 0 otherwise;

R2 =1 if the pixel is fire-coloured, and 0 otherwise;

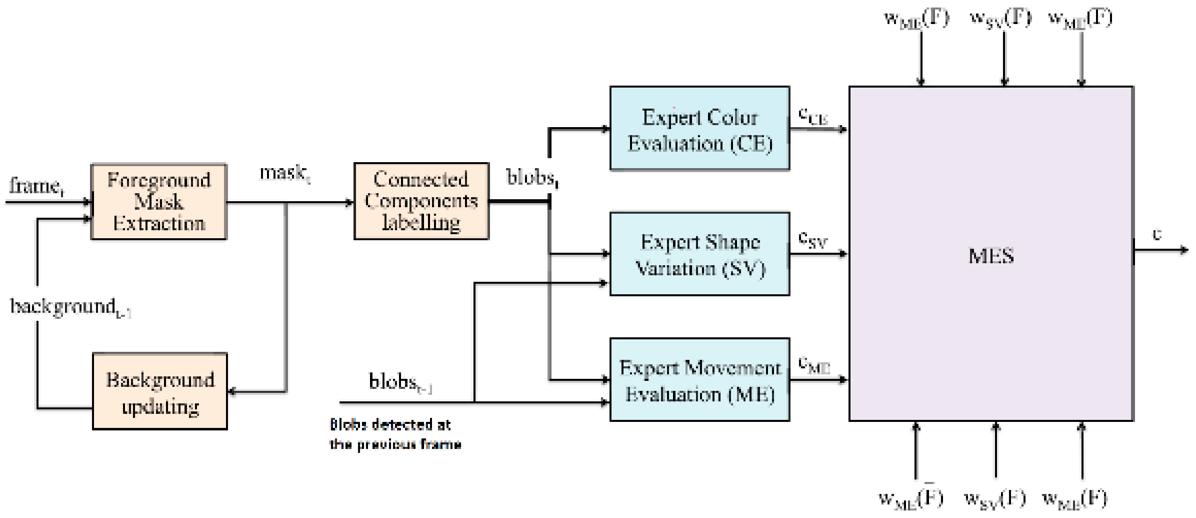
R3 = 1 if the number of zero crossings of e_n (x, y) and/or d_n (x, y) in a few seconds exceeds a pre-set threshold value, and 0 otherwise; and

R4 = 1 if the total energy of the wavelet details sub-images exceeds a threshold value, and 0 otherwise.

The voting decision $\begin{cases} IF\ \sum_i R_i \cdot w_i \ > \mathrm{T} & Then\ the\ pixel\ represents\ real\ fire \\ \\ ELSE, & the\ pixel\ represents\ non-fire \end{cases}$ …..(57)

Where w_i, T are weights and pre-set threshold defined by the users. The authors claimed a drastic reduction in false alarms by using the temporal and spatial wavelet analysis. This method can be used for early detection of fire in real-time, as well as in video databases. However, it has two disadvantages; first it is computationally demanding because of the two wavelet examinations; temporal and spatial [35]. Second, it cannot detect small fire regions.

Foggia et al (2015) presented another real-time fire detection approach, the authors implemented a multi-expert system (MES) to evaluate colour, variation of shape, and flame movements information, Figure 2.17. The final decision is taken by combining the individual decisions of each expert [8].



$w_{ME}(F)$
$w_{SV}(F)$
$w_{ME}(F)$
frame$_t$
Foreground Mask Extraction
mask$_t$
Connected Components labelling
blobs$_t$
Expert Color Evaluation (CE)
c_{CE}
Expert Shape Variation (SV)
c_{SV}
Expert Movement Evaluation (ME)
c_{ME}
MES
c
background$_{t-1}$
Background updating
blobs$_{t-1}$
Blobs detected at the previous frame
$w_{ME}(\bar{F})$
$w_{SV}(F)$
$w_{ME}(F)$

Figure 2.17 Overview of The MES Algorithm. [8]

Background subtraction algorithm is used to detect the moving object by subtracting the background frame from the current frame. The background frame is properly updated to count for the changes in the scene during the day. Then, the blobs representing moving objects in the scene are acquired by connecting the labeled regions in binary frame.

First, the colour information expert (CE): fire region is detected based in compliance with all the following six colour-rules in YUV colour space. Consider m=1,2,3,…M, and n=1,2,3,….,N are the image width and height in pixels, the colour rules are give by Eq. 58 thru Eq.63, as follows:

Rule 1: $Y(m,n) > U(m,n)$ …………………………………………………………… (58)

Rule 2: $V(m,n) > U(m,n)$ …………………………………………………………… (59)

Rule 3: $Y(m,n) > \frac{1}{MXN}\sum_{m,n} Y(m,n)$ .. …………………………………………… (60)

Rule 4: $U(m,n) < \frac{1}{MXN}\sum_{m,n} U(m,n)$ ……… …………………………………………… (61)

Rule 5: $V(m,n) > \frac{1}{MXN}\sum_{m,n} V(m,n)$ ……… …………………………………………… (62)

Rule 6: $|V(m,n) - U(m,n)| \geq 40$ ………… …………………………………………… (63)

Second, the shape variation expert (SV): the fire region shape variation is detected by calculating the perimeter (P_t) and the area (A_t) of the minimum bounding contour encloses that fire region in the present frame. Then the blob shape ratio at that frame is given by Eq. 64, as follows:

$$r_t = \frac{P_t}{A_t} \quad \text{……………………………………………………………………………… (64)}$$

Next, shape variation is calculated as percentage of the absolute difference between the ratios at the current and previous frames as compared to the ratio at the present frame. This is given by Eq. 65, as follows:

$$sv_t = \frac{|r_t - r_{t-1}|}{r_t} \quad \text{……………………………………………………………… (65)}$$

The blob shape is considered varying if the ratio sv_t is greater than a user defined threshold τ_v. The blob shape variation is given by Eq. 66, as follows:

$$\begin{cases} IF\ sv_t > \tau_v & Then\ the\ blob\ shape\ is\ varying \\ ELSE, & the\ blob\ shape\ is\ not\ varying \end{cases} \quad \text{……………………………..(66)}$$

Third, motion evaluation expert (ME): naturally, in case of uncontrolled fire, the tongue of flame moves randomly in different directions at the same moment. Conversely, the pixels representing rigid fire-coloured moving objects are moving in very limited directions. In order to interpret this observation, a new descriptor based on bag of words algorithm is implemented to characterize the cluttered motion of the flame boundary as follows:

1. Extraction of the low-level features: salient corner points of the blob at the present frame are extracted with their descriptors using SIFT algorithm.
2. The extracted features are matched with their ancestors in the previous frame.

Direction of each corner point is determined by comparing its current and previous positions, see the red and blue circles in .

Figure 2.18 a, b.

Create dictionary of words (probabilities of directions) to estimate and evaluate the direction of the motion, the space of all possible words is the angles between 0° to 360°. To limit the number of words and reduce the noise error, the space of all angles is clustered to six equal partitions, each partition is 60° span. The histogram distribution of the motion directions of the corner points in the blob is plotted versus the below bins, .

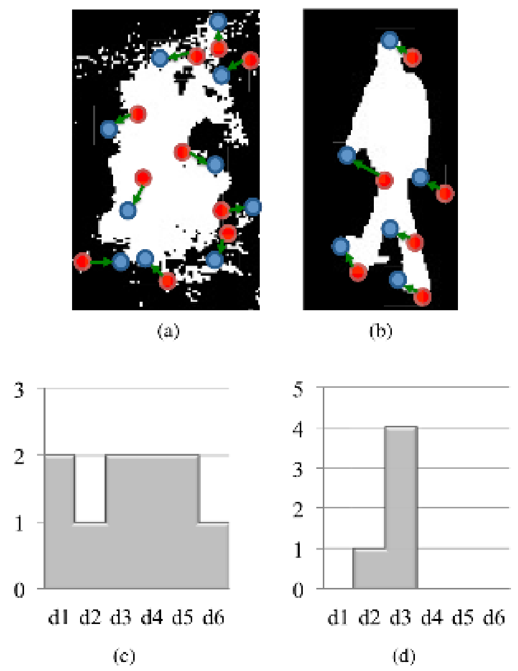
Figure 2.18 c, d.

Partition d1	Partition d2	Partition d3	Partition d4	Partition d5	Partition d6
0° to 60°	60° to 120°	120° to 180°	180° to 240°	240° to 300°	300° to 360°

3. Decision: the obtained histogram is considered a strong clue about the cluttered motion of the flame as compared to the uniform motion of fire-coloured moving objects like human body. In case of fire, the histogram distribution is evenly distributed on all partitions. Whereas the histogram representing rigid object movement looks like spikes on one or few partitions. To measure the distribution, the histogram homogeneity is calculated and compared to predefined threshold as given by Eq. 67 and Eq. 68, as follows:

$$\text{hm} = 1 - \frac{max(H)}{\sum_k h_k} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(67)$$

$$\begin{cases} IF\ hm > \tau_m & THEN\ the\ blob\ shows\ random\ motion \\ ELSE, & the\ blob\ does\ not\ show\ random\ motion \end{cases} \quad \ldots\ldots\ldots\ldots(68)$$



(a) Random motion of flame, (b) Uniform motion of rigid body,

(c) Histogram of the directions of the random motion,

(d) Histogram of the directions of the uniform motion.

Figure 2.18 Random motion of real fire pixels versus the uniform motion [8].

Finally, the decisions taken by the individual experts are combined by a MES classification model based on a weighted voting, which classify each blob as fire or non-fire. The weights are obtained from the precision of each expert algorithm in the training session. According to the authors, the algorithm has been tested on a wide database to assess its performance in terms of sensitivity and specificity. Experiment results show the effectiveness of the proposed algorithm, which supersedes the performance of its composing experts with huge reduction of false positives, from 29.41% to 11.76% and improvement in accuracy from 83.87% to 93.55%. Below is a tabulation for the obtained results using single and combination of experts:

Method		Accuracy	False Positive	False Negative
Single Expert	CE	83.87%	29.41%	0
	ME	71.43%	53.33%	0
	SV	53.57%	66.67%	21.85%
MES	CE + SV	88.29%	13.33%	9.74%

	CE + ME	92.86%	13.33%	0
	CE + ME + SV	93.55%	11.76%	0

However, this algorithm suffers from the following two disadvantages: first, expensive computations for the evaluation of the random motion (ME). The computation associated with ME alone consumes 85% of the total computation time [8]. Second, any fire coloured object moving towards (or away from) the camera will confuse the shape variation (SV) expert and causes false positive alarms, this is obviously noticeable from the above table of results.

Truong & Kim (2012) suggested a four-stage fire detection technique, starting by extracting the moving regions in a video sequence using adaptive Gaussian Mixture Model (GMM), then the fire colours of those regions is clustered using Fuzzy C-Means (FCM) algorithm. If the moving regions are fire coloured, then tempo-spatial features are extracted and passed to a support vector machine algorithm to classify the region as real fire or non-fire, Figure 2.19

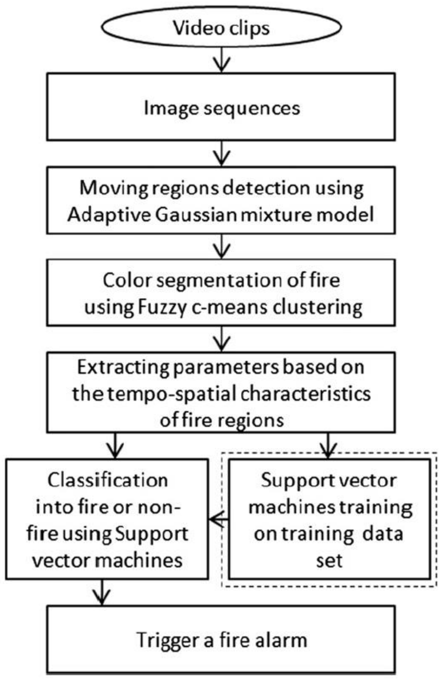

Video clips
Image sequences
Moving regions detection using Adaptive Gaussian mixture model
Color segmentation of fire using Fuzzy c-means clustering
Extracting parameters based on the tempo-spatial characteristics of fire regions
Classification into fire or non-fire using Support vector machines
Support vector machines training on training data set
Trigger a fire alarm

Figure 2.19 Flowchart of the proposed video-based fire detection technique

First, adaptive Gaussian Mixture Model (GMM) is used to model the background pixels and mask the moving region. This technique uses the first n-frames to learn about different values of each pixel in RGB colour space. The pixel values are represented by a number of gaussian probability distributions with their means are the pixel values that are repeated the most. In other words, the mean values represent the background colour for that pixel. Therefore, any pixel values beyond the variances will be considered a foreground or moving pixel. The technique is adaptive in the sense that the newer observed frames are heavily weighted than old frames to accommodate the changes in the observed scene. Based on experiments, the authors have chosen three gaussian distributions to model the background. Second, the image pixels are subjected to colour classification using soft clustering technique known as FCM algorithm as follows:

1. In CIE LAB colour space, the set of colour components A and B of all pixels are divided into two clusters: fire-colour and non-fire-colour. The two centers of the clusters are given some initial values (initialization). The initial values are computed empirically using sample fire images, Figure 2.20. The CIE LAB colour space is chosen for colour segmentation because, unlike RGB colour space, it is device-independent and separate luminance (illumination) information from chrominance (colour) information.

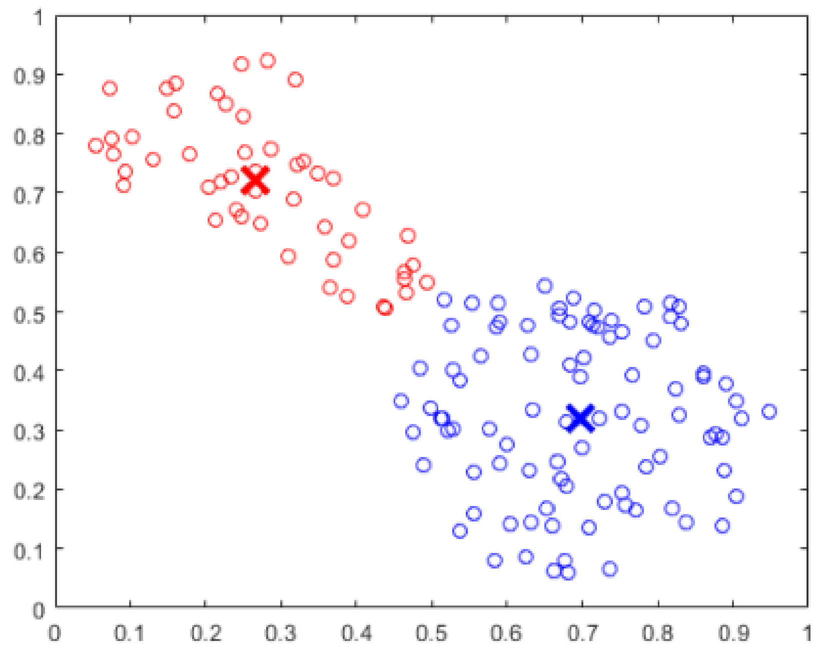


Figure 2.20 FCM Clustering. [17]

2. Then, the algorithm assigns degree of membership (probability) for each pixel colour values (A, B) to each cluster based on distance between the cluster center and that pixel values [18], Figure 2.20. The closer the pixel values be to the center of the cluster; the higher will be its degree of membership towards that particular cluster. The membership values are normalized

so that the sum of membership degrees of each pixel values to all clusters should be equal to one. For example, the pixel colours can be fire-colour to a degree of 0.85 versus 0.15 being non-fire colour.

3. After completing the clustering of all pixels, the centers of the clusters and the degrees of membership are updated iteratively according to the following equations:
   - Cluster center (mean) is given by Eq. 69, as follows:

     $$C_j = \sum_{i=1}^{n} x_i \cdot (\mu_{i,j})^r \ / \ \sum_{i=1}^{n} (\mu_{i,j})^r \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \ (69)$$

     Where $\mu_{i,j}$ is the degree of membership of pixel (x_i) to cluster (j) , C_j is the center (mean) of cluster j, n is the number of pixels, r is a fussiness index defined by the user, $r > 1$. The authors have chosen $r = 2$ empirically.

   - Distance $d_{i,j}$ from each pixel (x_i) to cluster center (C_j). $d_{i,j}$ is given by Eq. 70, as follows: $\left(d_{i,j}\right)^2 = ||x_i - C_j||^2$ …………………………………… (70)
   - $\mu_{i,j}$ is the degree of membership of pixel (x_i) to cluster (j). $\mu_{i,j}$ is given by Eq. 71, as follows: $\mu_{i,j} = 1 \ / \ \sum_{k=1}^{M} \left[\left(\frac{d_{i,j}}{d_{i,k}}\right)^2\right]^{\frac{1}{r-1}}$ …………………………… (71)

     Where, M is the number of clusters. In this calculation $M = 2$ clusters; fire colour and non-fire colour.

4. To ensure that all pixels are clustered correctly according to their colour values, the objective of FCM is to minimize the sum of all pixel distances to the cluster centers multiplied by the corresponding degrees of memberships to those clusters. The objective function of FCM is given by Eq. 72, as follows:

   Minimize $\sum_{i,j} ||x_i - C_j||^2 \cdot (\mu_{i,j})^r$ for all pixels x_i and all clusters j …………… (72)

5. The algorithm will be terminated when there is no further improvement to the cluster centers. Mathematically, this is given by Eq. 73, as follows:

   $$Max\ ||C_j{}^t - C_j{}^{t-1}|| \leq 0.001 \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ (73)$$

6. Finally, all moving pixels are assigned to a cluster according to the corresponding maximum membership values. If there is no center obtained for the fire-colour cluster, it can be

concluded that the objects in the moving regions are not fire. Conversely, the extracted fire-coloured moving regions are subjected to further processing in the next step in order to improve the robustness of the fire detection algorithm.

Third, the evolving of the candidate region is monitored over a sequence of frames, and seventeen (17) tempo-spatial statistical features are extracted to confirm compliance with the fire criteria, those features are:

1. Mean (M_A) and variance (V_A) that capture the random changes of the candidate region area;
2. Mean (M_I) and variance (V_I) that capture the random changes of the luminance (brightness) of the candidate region in grayscale;
3. Hu invariant moments [12]: a set of seven statistical parameters calculated using central moments that are invariant to image translation, scale, and rotation. Hu moments capture the roughness and coarse nature of the shape of the candidate region, and denoted by $[h_1, h_2, h_3, h_4, h_5, h_6, h_7]$; and
4. Motion templates that track the characteristics of the motion of the candidate region and extract six features, these are the magnitude and orientation $[M_v, O_v]$ of the motion, the mean of magnitude, the mean of orientation, the variance of magnitude, and the variance of orientation $[M_M, O_M, V_M, V_O]$.

Finally, all extracted features are concatenated in a (17X1) feature vector which is passed to a trained Support Vector Machine (SVM) classification to classify the candidate region as fire or non-fire. The SVM model is trained using training dataset collected from several video clips of fire, non-fire, or moving objects in different scenes under different illumination conditions. The Authors claimed that the proposed algorithm gives high detection accuracy for both the indoor and outdoor test video clips. Showing an average true positive (TP) of 94.78% and an average true negative (TN) of 98.68%. However, the algorithm suffers from the following disadvantages: first, complex and redundant computations that require longer computational time (1.486 seconds/ frame) [36]. This is a long time compared to the capabilities of recent surveillance cameras that can capture up to 30 frames per second. Therefore, a powerful and expensive hardware is required to perform the heavy computation. Second, it is not suitable for the real-time fire detection because of the long processing time.

Gong et al (2019) proposed an algorithm for flame detection in video sequence based on multi-feature fusion. The use of multi-features aims to reduce the high false alarm rate in other image-processing based fire detection algorithms. The algorithm starts with extracting the fire candidate region using the motion and colour information. Then spatiotemporal statistical features of that region are calculated and input to SVM algorithm to classify the region as fire or non-fire.

First, the moving region is extracted using a modified frame difference algorithm. It is simple to implement with low computation cost, and it adapts to scene changes and dynamic environments. Frame difference formula is given by Eq. 74, as follows:

$$F_v(m,n) = |F_c(m,n) - F_p(m,n)| \quad \ldots\ldots (74)$$

Where, $F_c(m,n), F_p(m,n), F_v(m,n)$ are the pixel-wise grayscale brightness of the current frame, the previous one, and the frame difference, respectively. The obtained frame difference is subjected to denoising and binarization through thresholding as shown in Eq. 75, as follows:

$$F_t(m,n) = \begin{cases} 255 \quad if \;\; F_v(m,n) \; \geq 20 \\ \\ 0 \qquad\qquad otherwise \end{cases} \quad \ldots\ldots (75)$$

The number 255 represents the white colour; the max brightness can be achieved for 8-bit image, and 0 represents the black colour. $F_t(m,n)$ is a black and white image, also known as binary image. Then $F_t(m,n)$ is subjected to a morphological dilation with a kernel size (m_1, n_1) to fill any holes in the extracted moving region. Therefore, all the nearby small regions are combined in single large moving region to simplify the computations. In the dilation process the kernel convolutes across the image, finding the local maximum brightness at each position and assign it to all pixels bounded by that kernel the morphological dilation is given by Eq. 76, as follows:

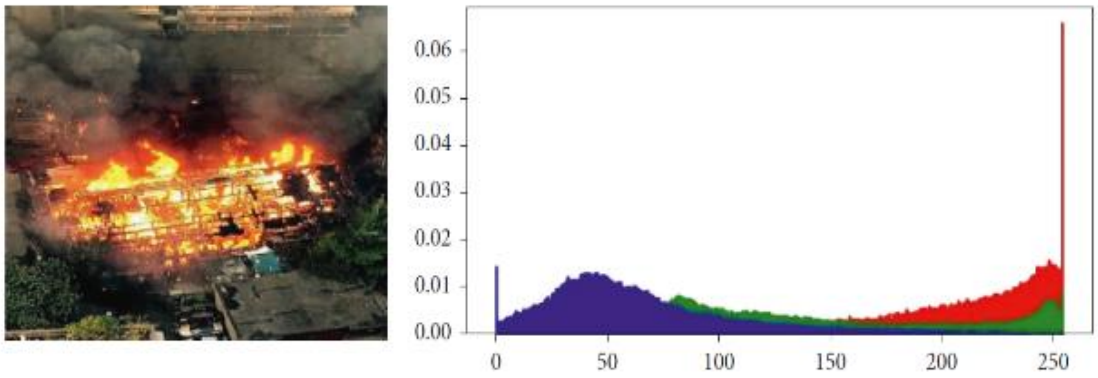
$$F_d(m,n) = Max(F_t(m + m_1\,,\; n\; + n_1)\,) \quad \ldots\ldots (76)$$

Second, the colour detection algorithm is applied to the obtained moving region based on experiential colour rules in RGB and HSI colour spaces. The colour rules are expressed in Eq. 77 thru Eq. 79, as follows:

$$\text{Rule 1:} \;\; \text{Red channel } (R) \; \geq \; 150 \quad \ldots\ldots (77)$$

Rule 2: $\text{Red channel } (R) > \text{Green channel } (G) > \text{Blue channel } (B)$ ……….………. (78)

Rule 3: $Saturation\ value\ (S)\ \geq\ 0.2$ ……………………………..……….………. (79)



(a) the flame image (b) RGB colour histogram

Figure 2.21 The flame image and the corresponding RGB colour histogram. [9]

Rule 2 is explained as shown in Figure 2.21 (b), the horizontal axis represents the pixel's colour values in RGB colour space, ranging from 0 to 255 for each channel (R, G, B), and the vertical axis shows the number of pixels on each value (the frequency). It is clear that the red channel pixel value is greater than the green channel pixel value, and the green channel pixel value is greater than the blue channel pixel value.

Applying both colour and motion detection algorithms gives the fire candidate region. The candidate region can be a real fire or just fire-coloured moving objects. Next step is to extract the statistical spatiotemporal features of the fire region for further verification. Third, the shape irregularity, the region growth, region motion, and shape changes are examined as follows:

1. Examine shape irregularity: Convex Hull algorithm [1] is applied to test the irregularity of the fire boundary shape. Convexity is defined as: a set $\mathbb{S}$ is convex if for any two points $p, q \in \mathbb{S}$ the line segment $p, q \subset \mathbb{S}$. For a set of points on a two-dimensional plane, like the fire candidate region, the convex hull is drawn by connecting the outermost points. The convex hull contains all points in the points set, Figure 2.22.

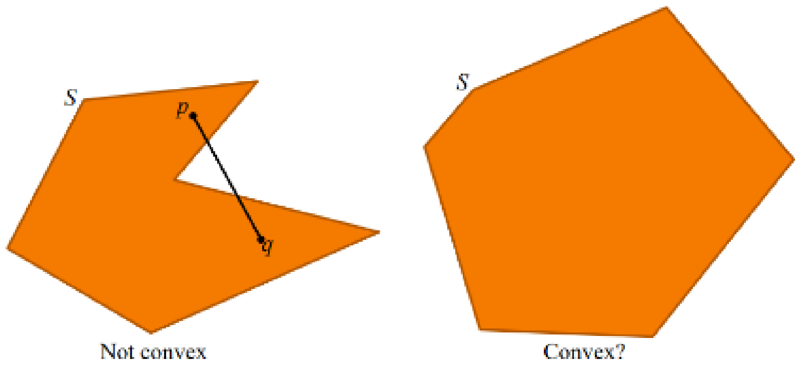


Figure 2.22 Nonconvex (concave) shape to the left versus Convex shape to the right.

Comparing the perimeter of the candidate region with the perimeter of the drawn convex hull is considered a clue about the regularity of the boundary of the region: $ratio, \mathrm{r} = \frac{P_{CH}}{P_{CR}}$, where P_{CH}, P_{CR} are the perimeters of the convex hull and candidate region respectively. The closer the two perimeters to each other (r is close to 1) indicates a regular boundary region and vice versa. Knowing that the uncontrolled fire region should have a non-convex (irregular) boundary, this test will eliminate all fire-coloured moving regions with regular shapes.

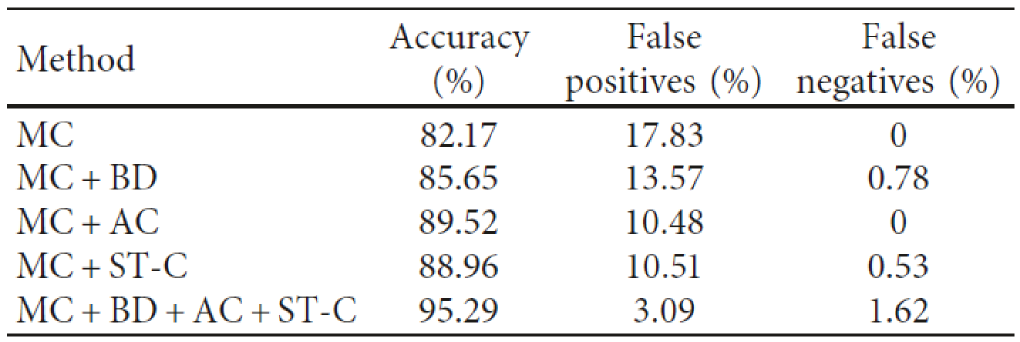
2. Examine the region growth: calculate the difference in the region area in two successive frames. Consider R_v is the ratio of the current change in the region area compared to the previous area. Higher R_v is considered a clue that the candidate region represents an evolving fire. The ratio of the region area change is given by Eq. 80, as follows:

$$R_v = \frac{|A_{n+1} - A_n|}{A_n} \quad \text{(80)}$$

3. Examine the region motion and shape changes: track the spatial centroid of the candidate region over number of successive frames. Spatial centroid of fire-coloured moving objects (vehicle or people) shows wide range of motion. On the other hand, fire region centroid is relatively stable as the core of fire is almost stationary. Assume d_{sum} is the Euclidean distance between two positions of the region centroids in two successive frames, $(x_i, y_i), (x_c, y_c)$. Smaller d_{sum} (less than a pre-set threshold) is considered a clue that the candidate region is fire. The sum of the spatial differences for all centroids over N number of frames is given by Eq. 81, as follows:

$$d_{sum} = \sum_N \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \quad (81)$$

4. In order to improve the system robustness and reduce the rate of false alarms, the extracted features are fused into a single feature vector as follows: $[r, R_v, d_{sum}, R, G, B, S, I]$ where r, R_v, d_{sum} are defined above, and R,G,B,S,I are the colour components in RGB and HSI color spaces. The vector is manually labelled as (fire and non-fire) and input to a trained support vector machine (SVM) algorithm for final verification. 77% of the fire data videos was used for training and 23% of the data was used for testing.



Method	Accuracy (%)	False positives (%)	False negatives (%)
MC	82.17	17.83	0
MC + BD	85.65	13.57	0.78
MC + AC	89.52	10.48	0
MC + ST-C	88.96	10.51	0.53
MC + BD + AC + ST-C	95.29	3.09	1.62

MC = Motion (frame difference) and colour detection

BD = Boundary detection (convex hull)

AC = Area change detection (change of number of pixels)

ST-C = Centroid change

Figure 2.23 Contribution of multi-features fusion to the detection accuracy.

According to the authors, the results proved that the feature fusion strategy has improved the detection accuracy and effectively reduced the false alarms, Figure 2.23. However, this approach suffers from the following limitations:

a) the complex computations require powerful hardware to run in real-time;
b) the convexity test is not reliable as it can be affected by the threshold values of the colour-segmentation and the illumination condition of the scene; and

c) the high rate of false alarms caused by non-fire but fire-coloured objects that are moving towards or away from the camera. Since such motion will confuse the area change (AC) and the centroid change (ST-C) calculations.

2.5.4. Smoke Detection

Naturally, smoke always occurs before the flame. Therefore, smoke detection will typically detect a fire quicker than flame or heat detection. However, smoke detection is not recommended where it may cause false alarms, especially in open environments, such as in the following cases:

- Outdoor because of the weather conditions like fog, rain, cloud, wind, or dusty environments may confuse the detection system and give false alarms.
- Where smoke, steam, or aerosol particles may spread out without fire because of the ongoing daily activities such as cooking, cleaning, maintenance works, construction works, running combustion engines, etc.

Regardless of the mentioned limitations, computer vision-based smoke detection may have specific applications. Some recent smoke detection algorithms are reviewed in this section.

Surit & Chatwiriya (2011) described an algorithm for smoke detection in video based on digital image processing by analyzing its static and dynamic features. Their idea is simple, smoke region can be distinguished by its gray colour (static feature). Smoke region covers a fixed position in the frame, and it is evolving, or in other words its area keeps changing over video sequence (dynamic features), the algorithm flowchart is shown in Figure 2.24.

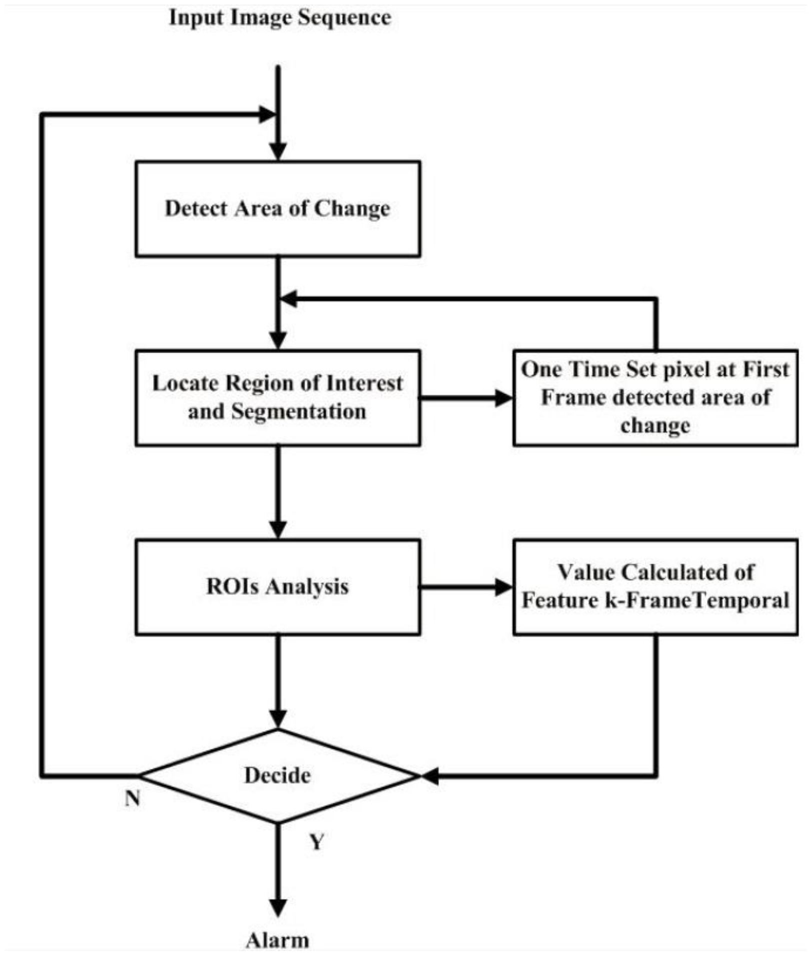

Input Image Sequence
Detect Area of Change
Locate Region of Interest and Segmentation
One Time Set pixel at First Frame detected area of change
ROIs Analysis
Value Calculated of Feature k-FrameTemporal
Decide
N
Y
Alarm

Figure 2.24 Flowchart of smoke detection algorithm. [31]

First, the area of change in the current frame is obtained by calculating the absolute difference between the current frame and the first frame. Frame difference is given by Eq. 82, as follows:

$$Im.Diff(t,m,n) = |ImBg(t,m,n) - ImCr(t,m,n)| \text{ ………………. (82)}$$

Then, Thresholding is applied on the frame difference image to get the binary image format. The threshold value ‘Th’ is a set to 0.1, and the thresholding is given by Eq. 83, as follows:

$$Im.Bin(t,m,n) = \begin{cases} 1 & IF\ \ ImDiff(t,m,n) > Th \\ 0 & otherwise \end{cases} \text{ ……………… (83)}$$

Second, determine the region of interest (ROI): connected component algorithm is applied to connect the segmented areas, followed by convex-hull algorithm to draw a polygon and rectangle contours enclosing the obtained region of interest. Then, calculate the colour (static characteristic) of the ROI. Average colour value at each pixel in RGB colour space is given by Eq. 84, as follows:

$$a_{m,n} = \frac{R(m,n)+G(m,n)+B(m,n)}{3} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (84)$$

Now, check whether the colour at each pixel in the ROI is gray. Consider R(m, n), G(m, n), and B(m, n) are the RGB color values at each pixel in the ROI, 'th' is a threshold. The gray colour check is given by Eq. 85, as follows:

$$\begin{cases} If\left|R(m,n) - a_{m,n}\right| \cap \left|G(m,n) - a_{m,n}\right| \cap \left|B(m,n) - a_{m,n}\right| < th \;\; Then\ color\ is\ \text{Gray} \\ Else \qquad\qquad\qquad\qquad\qquad\qquad color\ is\ not\ Gray \end{cases} \quad \ldots. (85)$$

Finally, the region is classified as smoke if the following three conditions are all true: if 1/4 of the number of pixels enclosed by the rectangle boundary are gray, and if the area enclosed by the polygon contour is changing over 30 consecutive frames, and if the area enclosed by the polygon contour keeps showing (i.e., it is a nonzero area over 30 frames). Else, the ROI is not a smoke region. This method uses threshold values to segment the gray smoke region. Thresholding segmentation is highly sensitive to the environmental and illumination conditions. Therefore, this approach is found not reliable for smoke detection at various conditions.

Cui et al (2008) used Wavelet Analysis and Gray Level Cooccurrence Matrices (GLCM) to extract the smoke texture feature from an image. Wavelet packet (Figure 2.25) is applied to decompose the image into multi-levels sub-images where the texture features are highlighted, g[n] is the low-pass approximation coefficients and h[n] is the high-pass detail coefficients. GLCM are then applied to quantify the texture features in these multi resolution sub-images into features vector using the statistical second order image moments. Finally, neural network classification algorithm is used to classify the features vector as smoke or non-smoke.

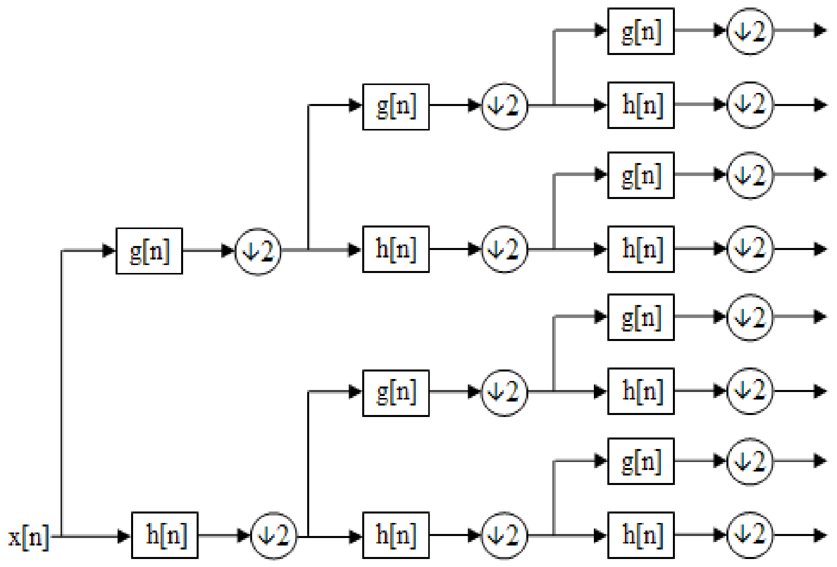


Figure 2.25 Wavelet Packet decomposition (WPD) up to 3 levels. [39]

The authors used Daubechies (D6) wavelet to compute two level wavelet packets. D6 wavelet has the advantages of high performance in texture classification and low complexity. The details sub-images of the two-level decomposition carry the high frequency details of smoke texture. Gray Level Cooccurrence Matrices (GLCM) is a statistical algorithm that quantifies the texture feature in each sub-image into a vector of features.

The elements of the GLCM correspond to the relative frequency of occurrence of two pixels in grayscale image, those two pixels having brightness equal to i, j, and they are separated by distance (d) and relative direction ($\emptyset$). Typically, d= 1,3,5, and 9, and $\emptyset = 0°$, 45°, 90°, and 135°. GLCM is denoted by: $p(i, j, d, \emptyset)$. Once the cooccurrence matrices (GLCM) are obtained, the following texture features can be calculated for each matrix; Entropy, Contrast, Angular Second Moment (ASM), Inverse difference moment (IDM).

Entropy; a measure for disorder in the image. Entropy is given by Eq. 86, as follows:

$$ENT = -\sum_i \sum_j p(i,j) \log[p(i,j)] \quad \text{(86)}$$

Contrast of the image is given by Eq. 87, as follows:

$$CON = \sum_i \sum_j (i-j)^2 p(i,j) \quad \text{(87)}$$

Angular Second Moment (ASM) also called image energy: It is a measure for uniformity of the image. ASM is given by Eq. 88, as follows:

$$ASM = \sum_i \sum_j [p(i,j)]^2 \quad \text{(88)}$$

Inverse difference moment (IDM): It is a measure for homogeneity of the image. IDM is given by Eq. 89, as follows:

$$\text{IDM} = \sum_i \sum_j \frac{1}{1+(i-j)^2} \cdot p(i,j) \quad \text{(89)}$$

Then the feature values are concatenated in a high-dimensional feature vector, this vector is input to two-layer back propagation neural network classification algorithm. The number of input nodes is equal to the number of elements (features) in the feature vector. The number of output nodes is two since it is a binary classification problem (i.e., the output is either smoke or non-smoke).

The authors claimed classification accuracy up to 98.1%, the accuracy varies depending on the number of inputs (the size of feature vector). In other words, it is a trade-off between the accuracy and the complexity of the computation. However, this method has a main disadvantage that, for accurate smoke detection, the smoke should cover large area of the image; hence it is not suitable for smoke detection at early stages.

Toreyin et al (2006) built their algorithm based on the fact that spreading of fire smoke causes the scene details to fade out. Moreover, as the smoke gets thicker, the colours values decrease. The image details are highlighted using spatial wavelet transform. The smoke detection is localized by dividing the image into smaller blocks, and watching the detail and colour fade out in each block.

First, for a video captured by a stationary camera, the moving pixels are extracted using the median background modeling. The background image is estimated as the average or median of the grayscale pixel values in the previous n frames, then the moving pixel is determined as the absolute difference between the current frame and the background model. To eliminate the effect of noise, this absolute difference should be greater than a specific threshold.

Second, those moving regions are analyzed using single-stage wavelet filter bank (Figure 2.26) to determine whether they are smoke or other moving objects. The image details such as edges and textures contribute to the high frequency coefficients that are highlighted in the wavelet detail sub-images. Sharp edges and corners in the image are translated to local extrema in those wavelet detail sub-images. Slow disappearing of the wavelet extrema and the decrease in the energies of those high frequency components in a sequence of frames is a clue about smoke spread.

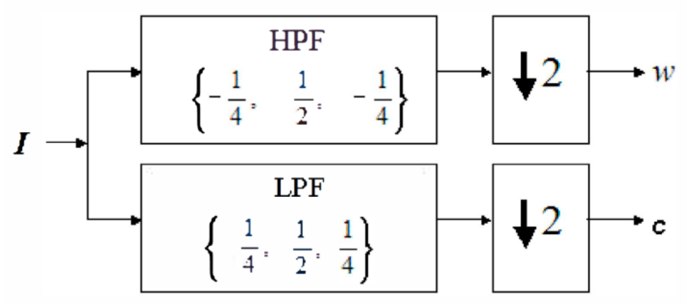

HPF
$\{-\frac{1}{4}, \frac{1}{2}, -\frac{1}{4}\}$
LPF
$\{\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\}$
I
↓2
↓2
w
c

Figure 2.26 Single-stage wavelet filter bank. [34]

The composite energy of detail sub-images of grayscale frame (n) is given by Eq. 90, as follows:

$$E_n(m,n) = |HL_n(m,n)|^2 + |LH_n(m,n)|^2 + |HH_n(m,n)|^2 \text{ .... (90)}$$

Where, $HL_n(m,n), LH_n(m,n),$ and $HH_n(m,n)$ are the horizontal, vertical, and diagonal wavelet detail sub-images respectively. Each sub-image is divided into small (k) blocks of size 8X8 pixels and the local energy value $\mathrm{e}_n^k(8,8)$ is calculated for each block.

Now, for the same block (k), if there is a gradual descent in the local energy value of the present frame (n) compared to the past frame (n-r), then this is considered a clue about smoke spreading in that block. Consider T1, T2 are two threshold values satisfy the condition $0 < T1 <$ T2 < 1. Then the gradual descent in the local energy is given b Eq. 91, as follows:

$$T1 \cdot \mathrm{e}_{n-r}^k(8,8) < \mathrm{e}_n^k(8,8) < T2 \cdot \mathrm{e}_{n-r}^k(8,8) \text{ .... (91)}$$

On the other hand, a quick drop in the energies or sudden disappearance of local extrema is interpreted as a rigid moving object entered the scene and blocking the view of the camera causing the details to completely disappear. The above thresholding condition is put to exclude the false alarm caused by strange moving object. Since $T1 > 0$, then $\mathrm{e}_n^k(8,8)$ will not drop to zero. In other words, the energy will not disappear completely in case of smoke spread.

Third, as the smoke evolves, it gets thicker causing the colours in the scene to be grayed out, reduction of the chrominance component values (U and V) in the YUV colour space of the present frame compared to the past frame is another clue about smoke spreading. YUV colour space is preferred over RGB space for the colour test because it has the advantage of separating the colour information (chroma) from the illumination information (luma), and hence eliminates the effect of shadows and changes in the illumination of the scene.

Fourth, knowing that the turbulent of uncontrolled fire flames flicker with a frequency of 10 Hz [34], [35], trained three-state Markov models are used to test the randomness of the temporal variation of the luminance value (Y_n) of pixels in the edges of the candidate region at frame n. Let $\mathcal{W}(n)$ be the wavelet coefficients of the edge pixels, which are obtained by applying the filter bank shown in Figure 2.26 on the candidate region. A three-state hidden Markov models (Figure 2.27) are defined using two non-negative threshold values in wavelet domain (Th2 > Th1 > 0). The three states are as follows:

- The State is in F1: $if\ |\mathcal{W}(n)| < Th1$
- The State is in F2: $if\ Th1 < |\mathcal{W}(n)| < Th2$
- The State is Out: $if\ |\mathcal{W}(n)| > Th2$

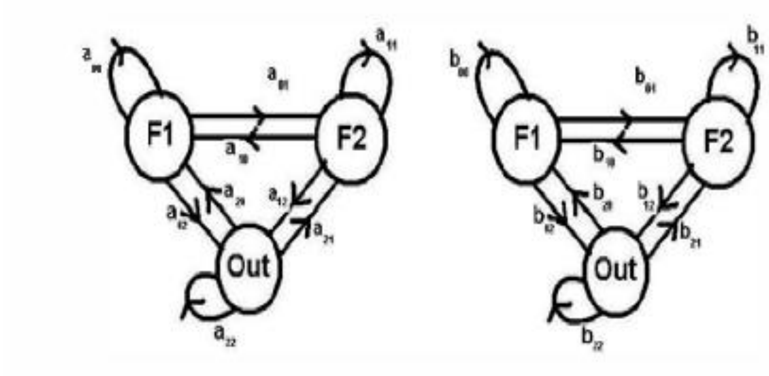


Figure 2.27 Three-state Markov models for smoke (left) and non-smoke pixels (right) [34]

The probabilities of transition between states for a pixel are estimated during a pre-determined time around smoke boundaries. The three-state Markov models are trained for both smoke and non-smoke pixels behaviour using the above states as features. After completing the training, two sets of probability parameters are estimated: a_{ij} for smoke, and b_{ij} for non-smoke, where both i and j = {1,2,3}. Because smoke boundary pixels are used for training, the model learns about smoke boundary flickers and spreading.

Finally, testing the randomness of the shape of smoke region is achieved by analyzing the contours of the candidate regions. Centroid of each region is calculated as explained earlier, then distances from that center to the boundary points can be obtained for all directions. In order to reduce the calculation complexity, the range $0 \leq \theta \leq 2\pi$ is quantified into 64 equally spaced angles (pins) and the distance from centroid to boundary point at each angle is calculated. The results are plotted below, Figure 2.28, for a smoke region on the top, and non-smoke region (vehicle headlight) on the bottom.

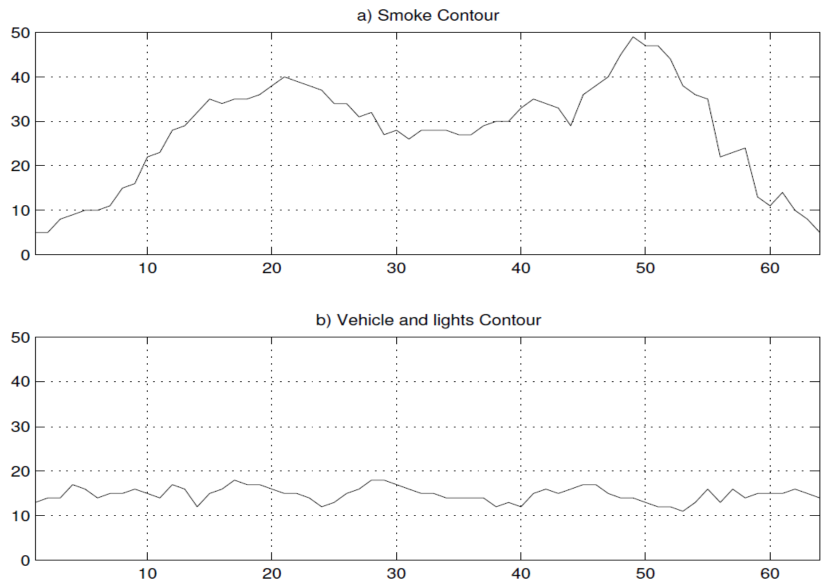


Figure 2.28 randomness of the shape of smoke region. [34]

It is obvious that the shape of smoke region exhibits more random variations and hence higher frequencies. To extract this higher frequency feature, one-dimensional wavelet filter bank is applied, and the ratio of the sum of high frequency coefficients ($\mathcal{W}$) to the sum of low frequency coefficients (C) is calculated as in Eq. 92 for smoke and non-smoke regions as follows:

$$\rho = \frac{\sum|\mathcal{W}(n)|}{\sum|C(n)|} \quad \text{.......} \quad (92)$$

Finally, the region is classified by comparing the ratio ρ to a threshold value. The classification is expressed in Eq. 93, as follows:

$$\text{The moving region} = \begin{cases} Smoke \quad IF\ \rho > Threshold \\ \\ Non\ smoke \quad otherwise \end{cases} \quad \text{.......} \quad (93)$$

According to the Authors, the algorithm successfully detected smoke in all sample videos with no false alarm. However, this method is expensive in computations and require powerful hardware to work in real time. Furthermore, the use of multiple threshold values increases the model sensitivity to the environmental and illumination conditions.

Yuan (2011) built a robust texture classification model based on the fact that smoke texture image is composed of uniform local patterns which, in turn, can be accurately described by image contrast and spatial statistical metrics. A multi-scale texture analysis technique is used to extract strong statistical features of the smoke pattern that are invariant to rotation and changes in illumination conditions. Local binary pattern (LBP) and Local binary pattern variance (LBPV) are the statistical features used in this algorithm. Then a trained neural network classification model is used to classify the image as smoke or non-smoke. First, set a 24X24 searching window to search for smoke textures. Then, a 3-level image pyramid is constructed, starting from grayscale 24X24 sub-image I_0, each level is a sub-sample from the previous level with a factor of two. The image pyramid is shown in Figure 2.29 (a), the flow chart of an image decomposition is shown in Figure 2.29 (b), the template of the low pass filter is shown in Figure 2.29 (c). The low pass filter is applied prior to down-sampling to minimize the noise effect.

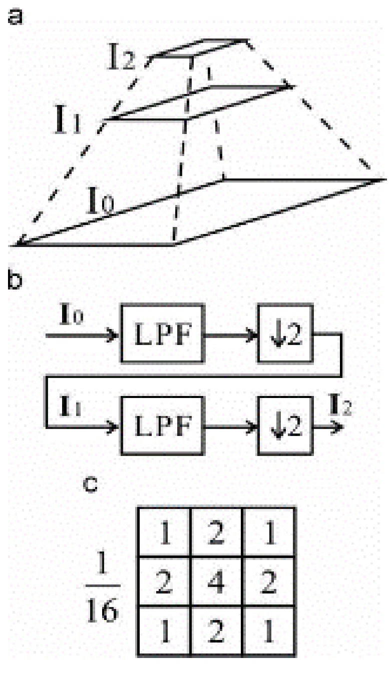
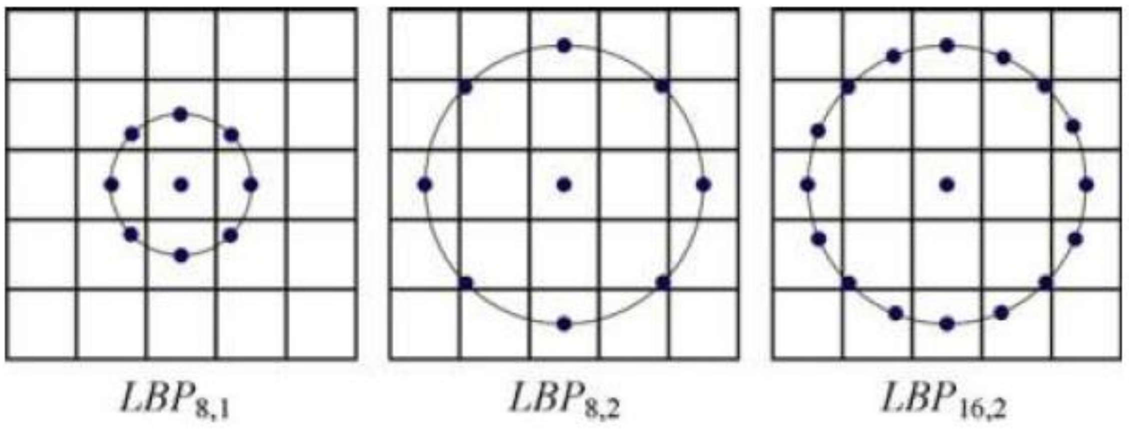


Figure 2.29 Pyramid decomposition of a grayscale image. [42]

Second, for each level image (I_0, I_1, and I_2), calculate the local binary pattern, LBP. LBP is a local pattern indicator that describes the relationship between a pixel and its neighborhood. LBPs can be computed as follows:

For each pixel, draw an imaginary circle centred at that pixel with radius *R*. The perimeter of the circle will pass by a number of *P* neighbour pixels surrounding the center pixel. Compare the brightness of the center pixel with each of the neighbour pixels, follow the pixels along a circle, either clockwise or counter clockwise. Whenever the center pixel is brighter than the neighbour pixel, then the LBP descriptor equals zero, otherwise, the LBP equals one. The 0/1 values are stored in a string of binary bits of length *P* having the notation: $LBP_{P,R}$, then this binary number is decoded to a decimal number which is assigned to the center pixel. Since the LBP is all about the comparison between the center and neighbour pixel values, then it is invariant to rotation and illumination conditions. For example, let the circle radius R= 1, then the number of neighbour pixels P=8, Figure 2.30 a, this is one block shift at each direction, this results in an 8-bit binary value, LBP8,1.



(a) R=1 and P=8 (b) R=2 and P=8 (c) R=2 and P=16

Figure 2.30 LBP descriptor for different radii and neighbour pixels. [42]

For small radius R, the LBP carries local pattern information, and for larger R the number of neighbour pixels increases, and the LBP carries global information. In this study, LBP8,1 is calculated for each level image (I_0, I_1, and I_2), and hence it carries both local and global pattern information. Consider B_c, B_j are the brightness of the center and neighbor pixels respectively. The LBP calculation is given by Eq. 94 and Eq. 95, as follows:

$$LBP_{P,R} = \sum_{j=0}^{P-1} \Phi(B_j - B_c) \cdot 2^j \quad (94)$$

$$\Phi(B_j - B_c) = \begin{cases} 1 & if\ (B_j - B_c) > 1 \\ 0 & otherwise \end{cases} \quad (95)$$

To eliminate the effect of noise and outliers, only the uniform LBP values are considered to form the vector of features. Uniform LBP means it has maximum of two bitwise transitions from 0 to 1 or vice versa. For example, the following LBP are uniform: 00000000, 11110000, 01111110, 00111100. Conversely, the following LBP are not uniform: 00110011, 11101110, and so on. This process is known as mapping from original pattern $LBP_{P,R}$ to a uniform pattern $LBP_{P,R}^{U2}$.

Third, the local binary pattern variance (LBPV) is calculated in a similar way. LBP variance (LBPV) is a texture descriptor that uses variance as an adaptive weight to adjust the contribution of the LBP descriptor in histogram calculation. While the local binary patterns carry the local and global pattern information, the local binary pattern variances carry the local contrast values, hence they are also known as local contrast values. Let M, N are the image width and height, the value of the kth bin of the LBPV histogram is expressed in Eq. 96 and Eq.97, as follows:

$$LBPV_{P,R}(k) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \omega\,(LBP_{P,R}(m,n), k) \quad (96)$$

Where:

$$\omega(LBP_{P,R}(m,n), k) = \begin{cases} VAR_{P,R}(m,n) & if\ LBP_{P,R}(m,n) = k \\ 0 & otherwise \end{cases} \quad (97)$$

So far, LBP and LBPV pyramids are computed for the image level (I_0, I_1, and I_2), Figure 2.31. Fourth, the histogram sequence is computed for the above six images. The histograms are denoted as $H_0^{u2}, HV_0^{u2}, H_1^{ri}, HV_1^{ri}, H_2^{riu}, HV_2^{riu}$. By the end of this process, the histograms are concatenated to form a single feature vector, F, as expressed in Eq. 98, as follows:

$$F = \left[H_0^{u2}, HV_0^{u2}, H_1^{ri}, HV_1^{ri}, H_2^{riu}, HV_2^{riu}\right] \quad (98)$$

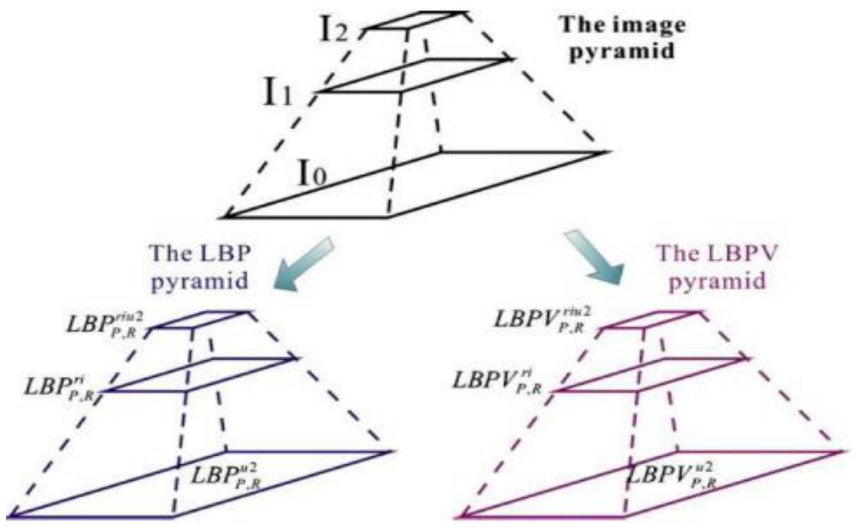


Figure 2.31 LBP and LBPV pyramids. [42]

Eventually, the obtained feature vectors (F) for different sample cases are manually labeled and used to train a multi-layer feedforward back propagation neural network classification model, Figure 2.32. The author used learning database composed of 1220 smoke images and 1648 non-smoke images. Out of which 552 smoke and 831 non-smoke images are used for model training, the reminder images are used for model testing and validation.

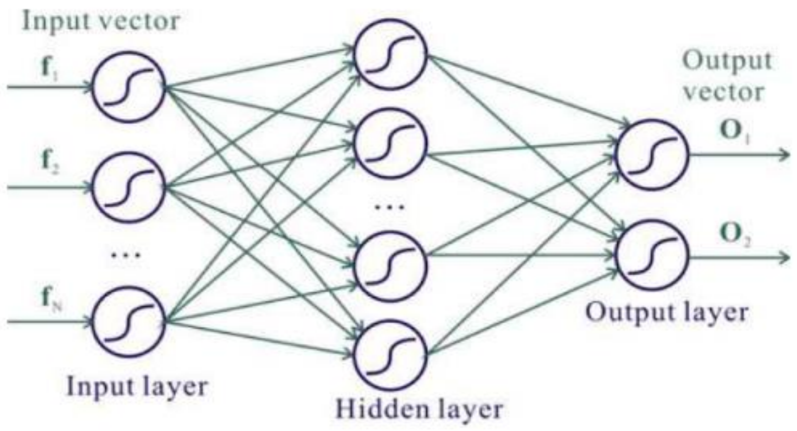


Figure 2.32 Smoke detection ANN. [42]

The reported results show that the model has a detection rate of 0.953488 for testing data images and 0.996377 for training data images, it is also shown that the misclassification rate was 0.033887 and 0.003615 for testing and training data images respectively. The misclassified images are those contain hazy objects such as water surface or unclear regions. The authors suggested using motion detection algorithm to eliminate those misclassified regions as much as possible. The proposed algorithm has the following limitations [42]:

1. The number of smoke pixels in the image should be large enough to be detected.
2. If a video contains too many objects which are not included in the training set, the system performance will drop significantly.
3. The algorithm can detect the presence of smoke in a 320x240 video at about 10 frames per second (fps). The algorithm cannot reach real-time processing for frame rates greater than 25fps. This limitation, in specific, makes the algorithm not suitable for real time applications since most of the surveillance cameras in the market capture videos at 30 fps.

2.5.5. Summary of The Research Gaps

The basic workflow adopted in most of the literature runs in a series of three stages: the first stage aims to extract an array of features to represent the fire region by applying image processing filters such as colour filters, wavelet, edge detection, Hu Moments for shape variations, and GLCM or LBP for texture detection. In the second stage, the obtained array of features is labelled and used to train either traditional machine learning (ML) classification algorithms such as Support Vector Machine (SVM) or modern ML classification algorithms such as artificial neural network (ANN). Lastly, the trained model is used to classify the fire candidate image as fire or non-fire. In a broad view, the reviewed algorithms can be classified into two groups:

1) Simple computations algorithms such as colour checking with basic motion detection, similar to [4], [24], [28], [29], [37], [38], [40]; and
2) Expensive computations algorithms such as spatiotemporal frequency analysis, pattern recognition calculations, and multi-feature fusion (i.e., manual features extraction), similar to [6], [8], [9], [16], [34], [35], [36], [42], [44].

To find the appropriate algorithm, software models for most of those algorithms have been implemented and tested using sample videos and images of a real fire and fire-coloured moving objects such as people with fire-coloured clothes, fire-coloured vehicles, flags, etc. Despite the easy computations, the results of the first group algorithms were disappointing; those colour and motion information only algorithms suffer from a major disadvantage; that is the high rate of false-positive alarms. Those algorithms cannot differentiate between a real fire and fire-coloured moving objects. The high rate of false alarms makes those simple-computations algorithms completely inappropriate and unreliable for early fire detection in urban areas: indoor and outdoor.

The expensive computations algorithms gave better results (compared to the first group), obviously because of the heavy computations used, but this comes at the cost of the processing time. Most of those algorithms require more than one second to test a single video frame, for example [36]. That is a very long processing time compared to the recording speed (25-30 fps). It is concluded that those expensive computations algorithms are either inappropriate for real-time applications or require expensive and powerful hardware to run in real-time.

Furthermore, those expensive-computations algorithms gave satisfactory results only in the favourable conditions; when the fire candidate lies on the focus of the camera and it is large enough to occupy almost the whole image/ frame, Figure 2.33 left, which is not always the case in real-life scenarios. Conversely, all the above methods have failed to classify small out-of-focus fire blobs, Figure 2.33 right. Misclassification of fire images /videos leads to a high rate of false alarms, lower accuracy, and less algorithm reliability.

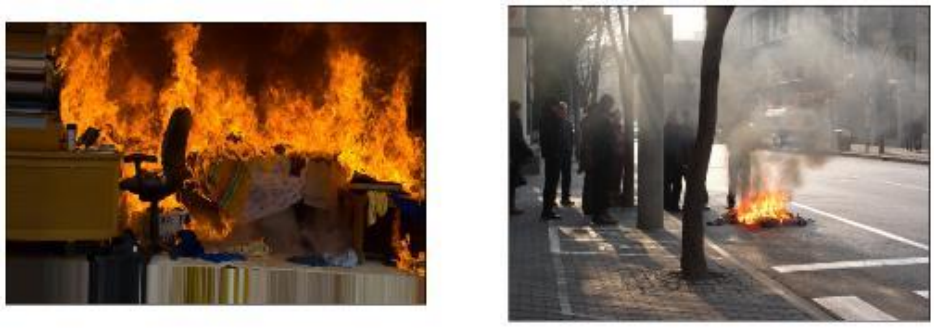


Figure 2.33 Large fire (left) vs small fire (right).

It has been found that image features such as colour segments, entropy, energy, contrast, wavelet coefficients, and Hu moments are insufficient for early and accurate fire detection, especially when the fire blob is small at the beginning of the fire incident. For early and efficient detection of small fire blobs, the value of each pixel in the image and its relation to the surrounding pixels have to be taken into consideration. This is achieved by implementing a Deep Learning Convolution Neural Network (DL CNN) algorithm to automate the process of feature extraction and use the extracted features to classify the image as fire or non-fire. The CNN algorithm has the following advantages over the expensive computations (manual feature extraction) algorithms:

1. The CNN algorithm performs the computations on the pixel level. Each pixel is represented by a separate neuron in the CNN model;
2. The CNN algorithm moves the heavy calculations to the offline training stage, i.e. all the expensive computations required to find the optimum network weights and biases are performed in the training phase before the real-time usage. Ultimately, the trained CNN model can run smoothly in real-time on average computer hardware;
3. The CNN algorithm automates the process of feature extraction, and it merges both feature extraction and image classification processes into one integrated operation; and
4. The CNN algorithms are learnable and can be continuously trained with new data to improve accuracy and minimize false alarms.

2.6. Summary

This chapter reviews some of the recently developed computer-vision based fire detection approaches. The review covered a wide range of fire detection algorithms: ranging from image processing fire detection based on colour information only, video processing fire detection based on colour and motion/growth information, advanced video processing fire detection based on colour, motion, and spatiotemporal frequency analysis, and lastly smoke detection algorithms. The advantages and limitations of each algorithm have been discussed. The review revealed that there is a room for further improvement and the research gap has been identified. There is a need for a high-accuracy fast-acting fire detection system that can detect the occurrence of small fire blobs early enough before becoming totally out of control and turning into a threatening risk to people and properties. Prototype of the proposed real-time flame detection model is demonstrated in detail in chapter 3.

Chapter 3. The Methodology of The Proposed System

3.1. Introduction

The chapter starts with discussing the efforts made to collect and prepare the video and image datasets for training, testing and validation of the proposed system. The image augmentation process is demonstrated. Then, this chapter provides an overview of the proposed two-stages early fire detection system and its main components. Later, this chapter presents detailed description of the stage-I of the proposed system, this is the video processing algorithm. This chapter ends by in-depth demonstration of the design of the stage-II of the proposed system, this is the CNN image classification model.

3.2. Data Collection and Preparation

A large number of videos and images have been collected from online libraries and online search tools like Google images. The collection has been selected to represent different real-life scenarios: indoor and outdoor, with and without fire, day and night, etc. The efforts to collect and prepare the sample videos and images are demonstrated below.

3.2.1. Video Samples

Sample video clips have been downloaded from the internet to test the performance of the proposed video processing algorithm. The downloaded videos are selected to be representing real-life indoor and outdoor scenes (with and without fire) recorded by generic surveillance cameras to simulate the real-life scenarios. Furthermore, to further challenge the algorithm's performance, the videos are selected to show fire-coloured moving and stationary objects (such as people with fire-coloured clothes, street lighting, fire-coloured vehicles, flags, furniture, etc.). Those videos are selected to be of various resolutions (480p, 720p, and 1080p) and different frame rates (25 and 30 frames per second).

3.2.2. Image Dataset

The image dataset is a crucial element of the modern ML image classification framework. Typically, the dataset is split into two parts: training subset, which includes around 75~80% of the overall dataset. Training subset used to train the weight and bias values of the CNN model in the model learning phase (training phase). The second part is the testing subset, which is the remaining

20~25% of the overall dataset. Testing subset used to test the model generalizability and measure its accuracy in the testing phase (validation phase).

The CNN image classification model is a supervised learning technique where the images in the training and testing datasets are manually labelled as fire images and non-fire images. A large and versatile dataset helps to build a robust and generalized CNN model which can be deployed in a wide range of different scenarios: indoor: in homes, residential, industrial, or institutional buildings and outdoor: in parks, streets, highways, construction sites, parking lots, airports, aircraft hangars and so on so forth. At the same time, the generalized model can give relatively high accuracy predictions for fire conditions day and night, in different weather conditions 247/365 days a year.

The fire dataset from Kaggle.com has been downloaded [13]. The dataset was created during the NASA Space Apps Challenge in 2018. The dataset size is 408.78 MB, and it contains 755 wildfire images and 244 non-fire images showing forests, trees, grass, river, lake, and waterfalls. That dataset is found inadequate for the objectives of this study, neither quantitively nor qualitatively.

From the quantity perspective, the total number of images in the dataset is less than one thousand images, which is insufficient to train a robust CNN model. More importantly, from the qualitative perspective, the images show wildfire and forested areas only; the fire flames cover the whole area of the fire images. There are no images to show fire/non-fire conditions in urban areas, cities, roads, vehicles, road signs, pedestrians with fire-coloured clothes, houses, or buildings. A couple of other datasets that have been used in previous studies are downloaded from the internet. Those datasets were found to be either lacking diversity or having large fire flames covering almost the whole images, see Figure 3.1, which represents a late-stage fire spreading. Such images are inadequate to train an early fire detection model that aims to detect fire conditions early and precisely.

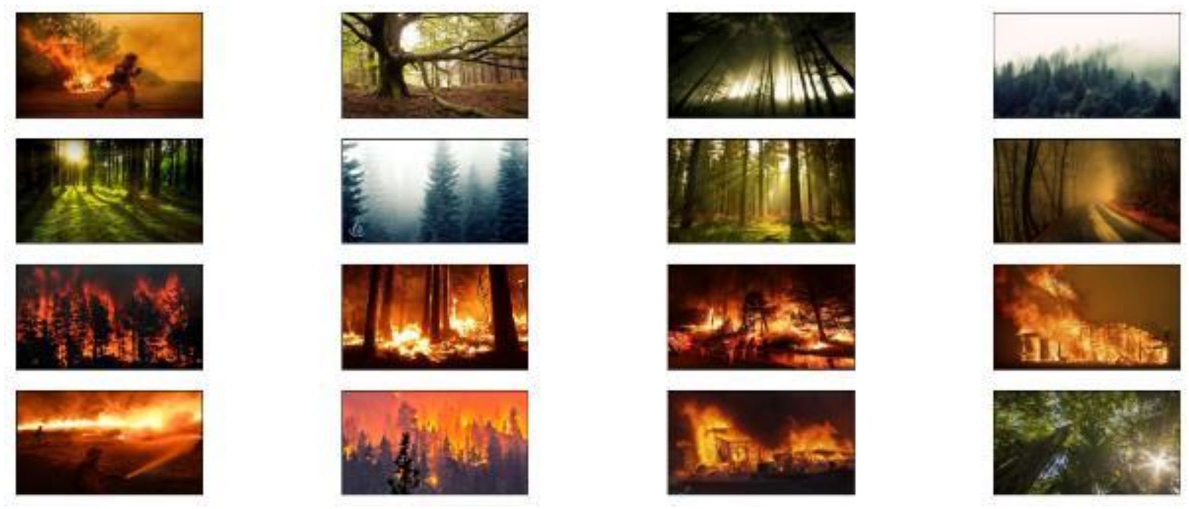


Figure 3.1 Sample of fire and non fire images of the dataset downloaded from kaggle.com.

Google Images is used to search the internet for more relevant images using keywords like fire in highway, home fire, high rise building fire, bus station fire, people with fire-coloured clothes, fire-coloured flags, fire-coloured home furniture, construction site warning signs, etc. The 8-scene image dataset [20] is also used. The dataset contains 8-natural scenes: streets, inside the city, tall buildings, highways, open country, mountain, forest, and coast as a natural accompaniment to the fire dataset as it shows natural scenes as they should look without fire present. More useful images are found on github.com, ultimatechase.com. Mivia image processing research Lab (www.mivia.unisa.it/) in The University of Salerno, Fisciano, Italy have been contacted to provide access to their fire dataset. The final dataset used to train/test the proposed CNN image classification model contains 5,300 images. This dataset is split into 75% training subset and 25% validation subset, **Error! Reference source not found.**. Sample images from the u nlabelled training subset are shown in Figure 3.3 below.

	Training dataset	Testing dataset
Number of Non-Fire images	2,008	500
Number of Fire images	2,292	500
Total	4,300	1,000

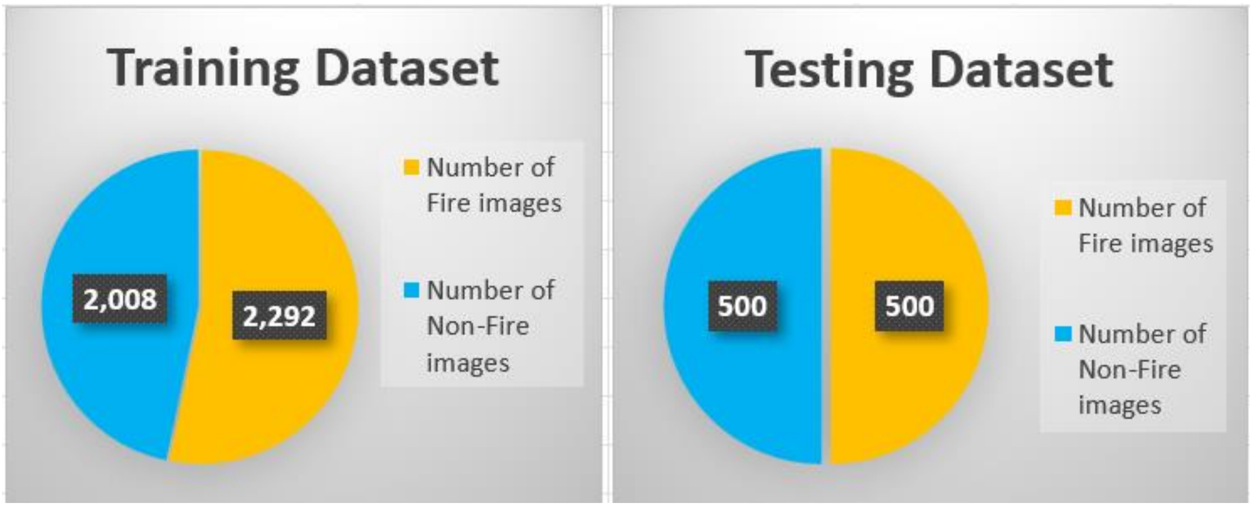


Figure 3.2 Images dataset is split into training and testing subsets.

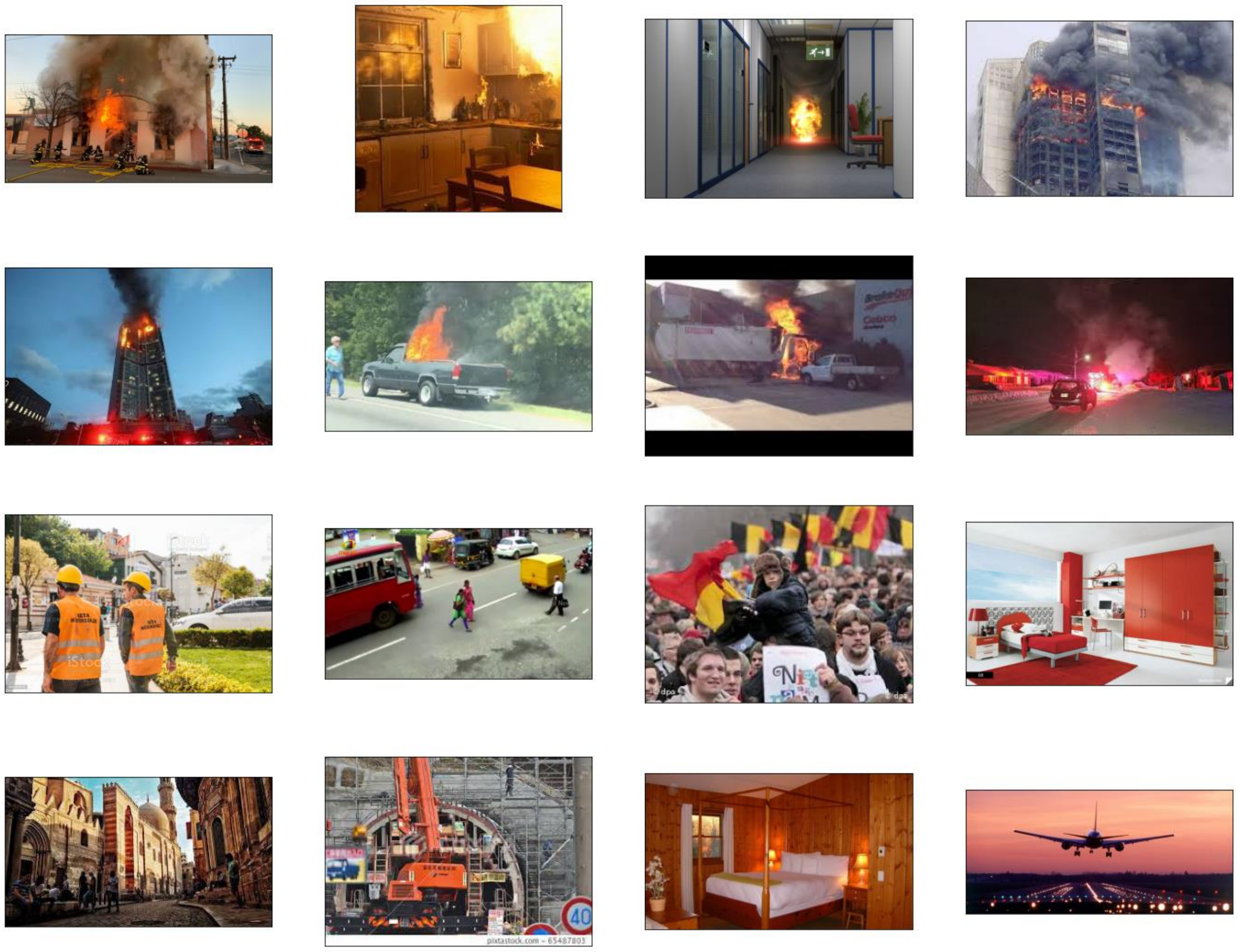
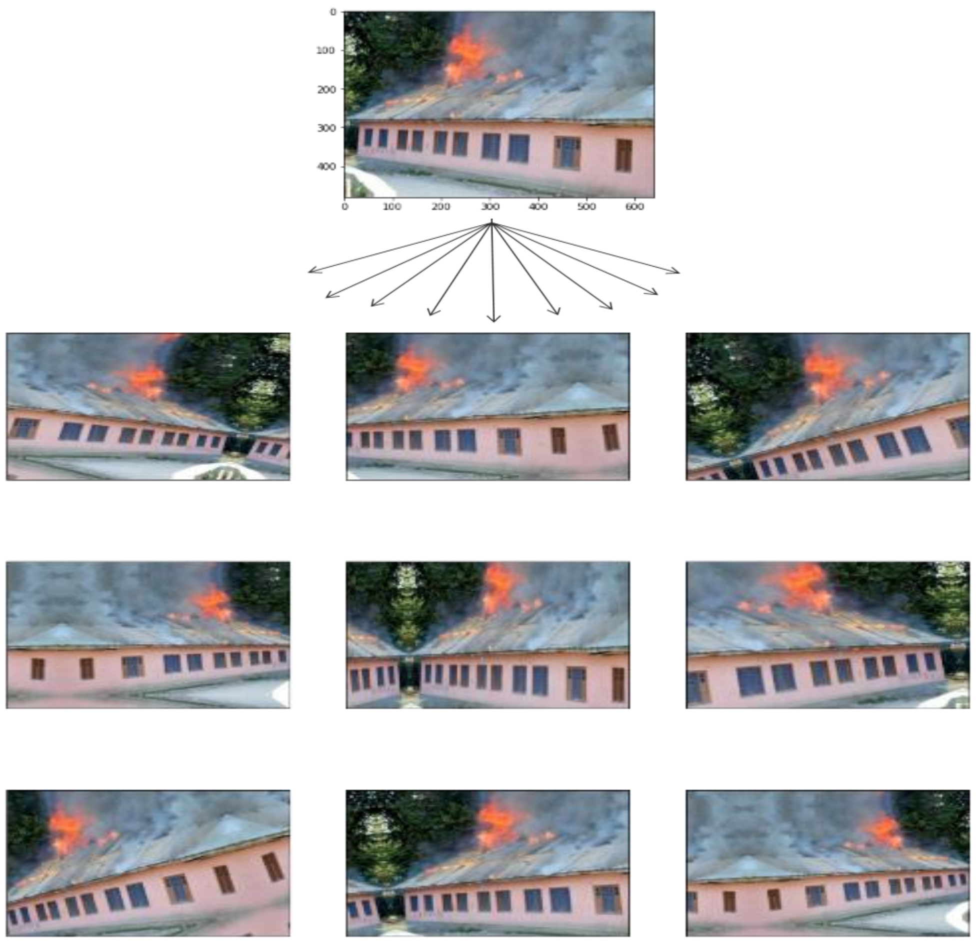


Figure 3.3 Sample of fire and non fire image dataset used to train the proposed CNN model.

3.2.3. Data Augmentation

The more data used to train the CNN classification model, the better is the model performance, and the higher is the classification accuracy. In the context of image processing, data augmentation is a technique to increase the number of training images virtually by cloning the original data images and applying random transformations on them. Such as rotation, shift, shear, change the brightness, and horizontal/vertical flipping to create modified versions from the original training images, Figure 3.4. The contents of the original images and the salient features will remain invariant and then can be extracted successfully for further processing. Data augmentation enriches the training data and leverages the model learning process. The Keras/TensorFlow deep learning library facilitates the use of data augmentation automatically when training the model. Also, the library gives the user the ability to set limits for the rotation angle, zoom level, and the ranges for shift and shear. Image normalization is performed along with the data augmentation.



Top: the original image, bottom: virtual augmented images.

Figure 3.4 Image data augmentation.

Finally, all images in the dataset are randomly shovelled and manually labelled, label-0 for non-fire images, while label-1 for fire images. Samples of augmented and labelled images from the training dataset are shown in Figure 3.5, Figure 3.6, Figure 3.7, and Figure 3.8.

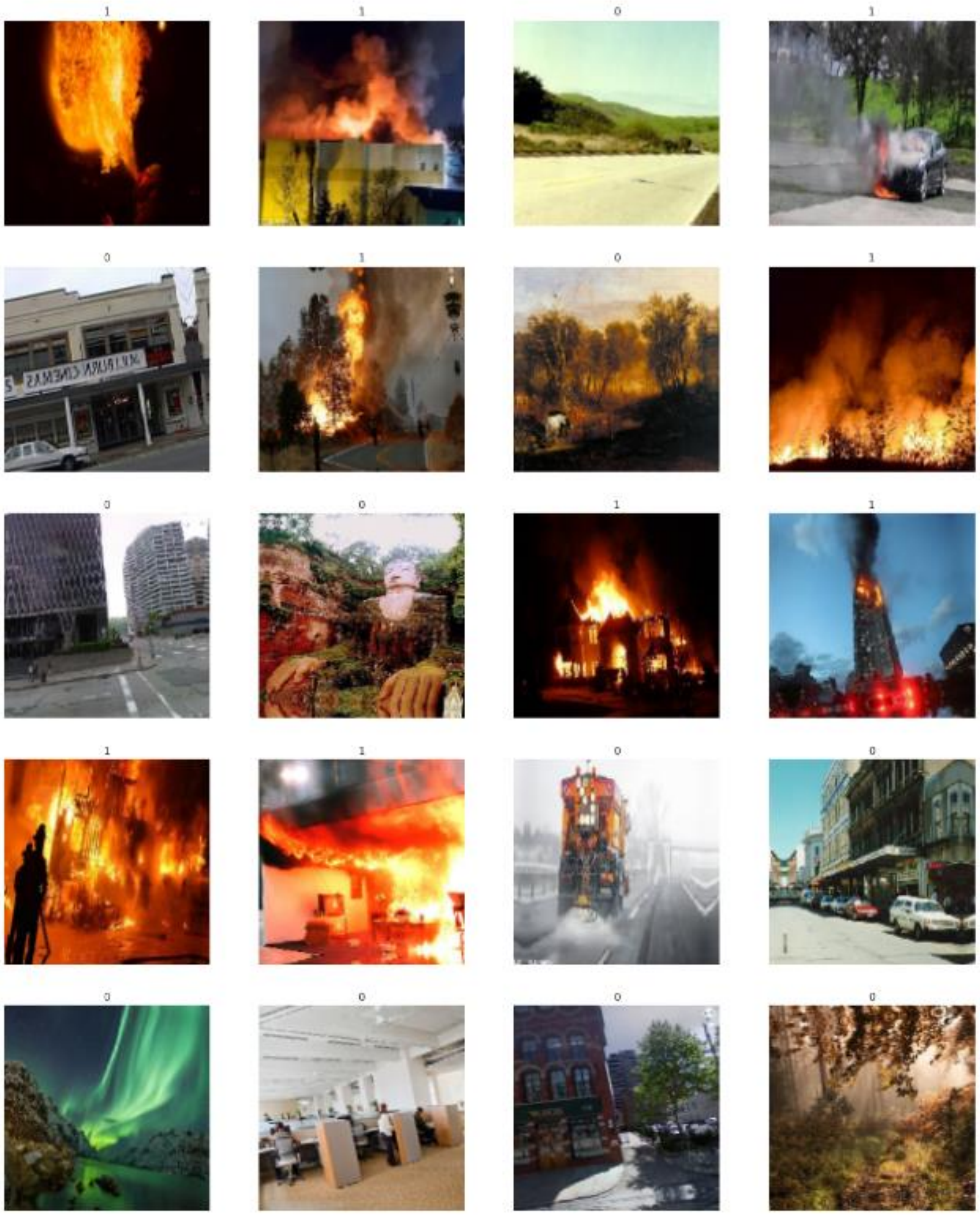


Figure 3.5 Sample-1 of labeled images from the training dataset

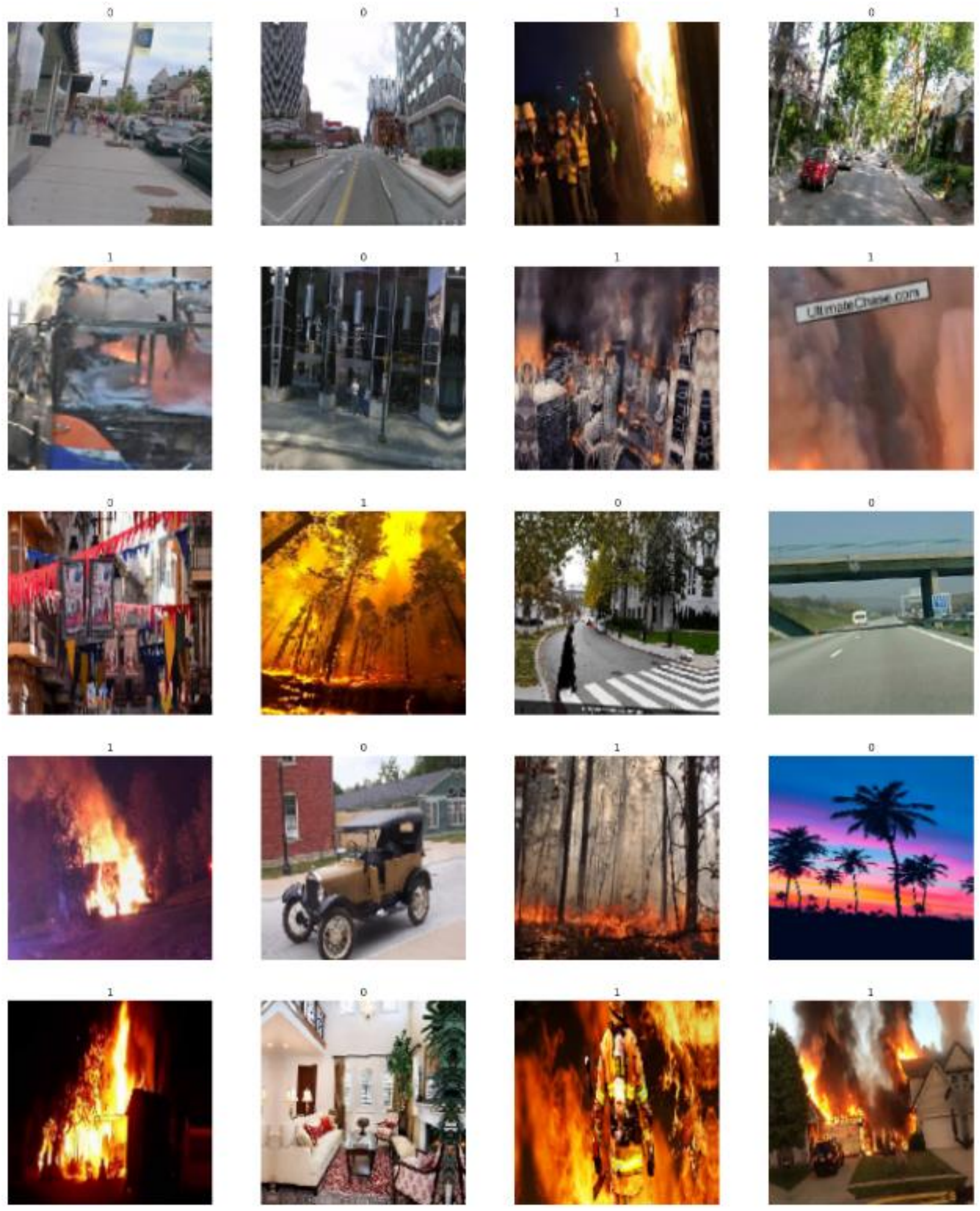


Figure 3.6 Sample-2 of labeled images from the training dataset

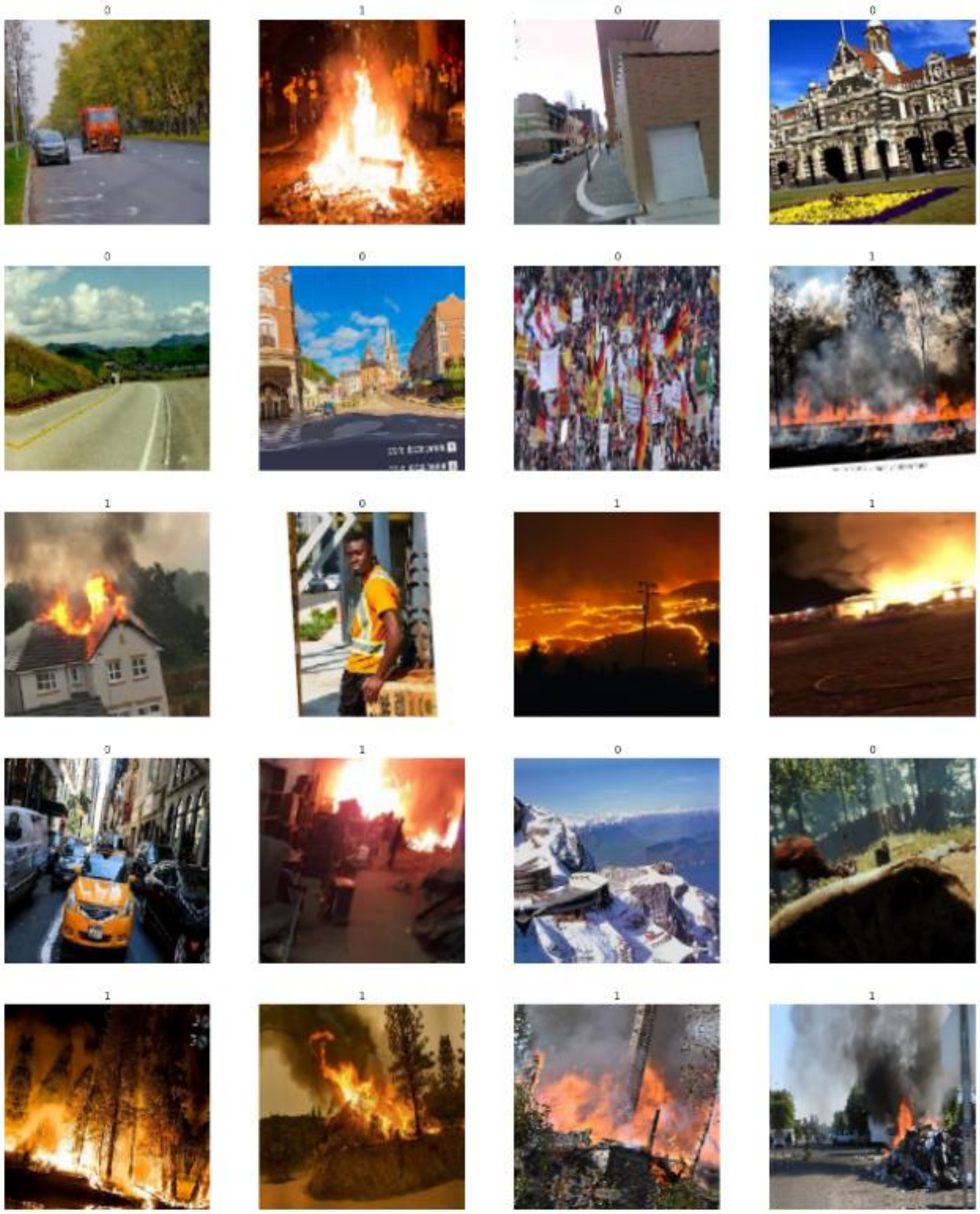


Figure 3.7 Sample-3 of labeled images from the training dataset

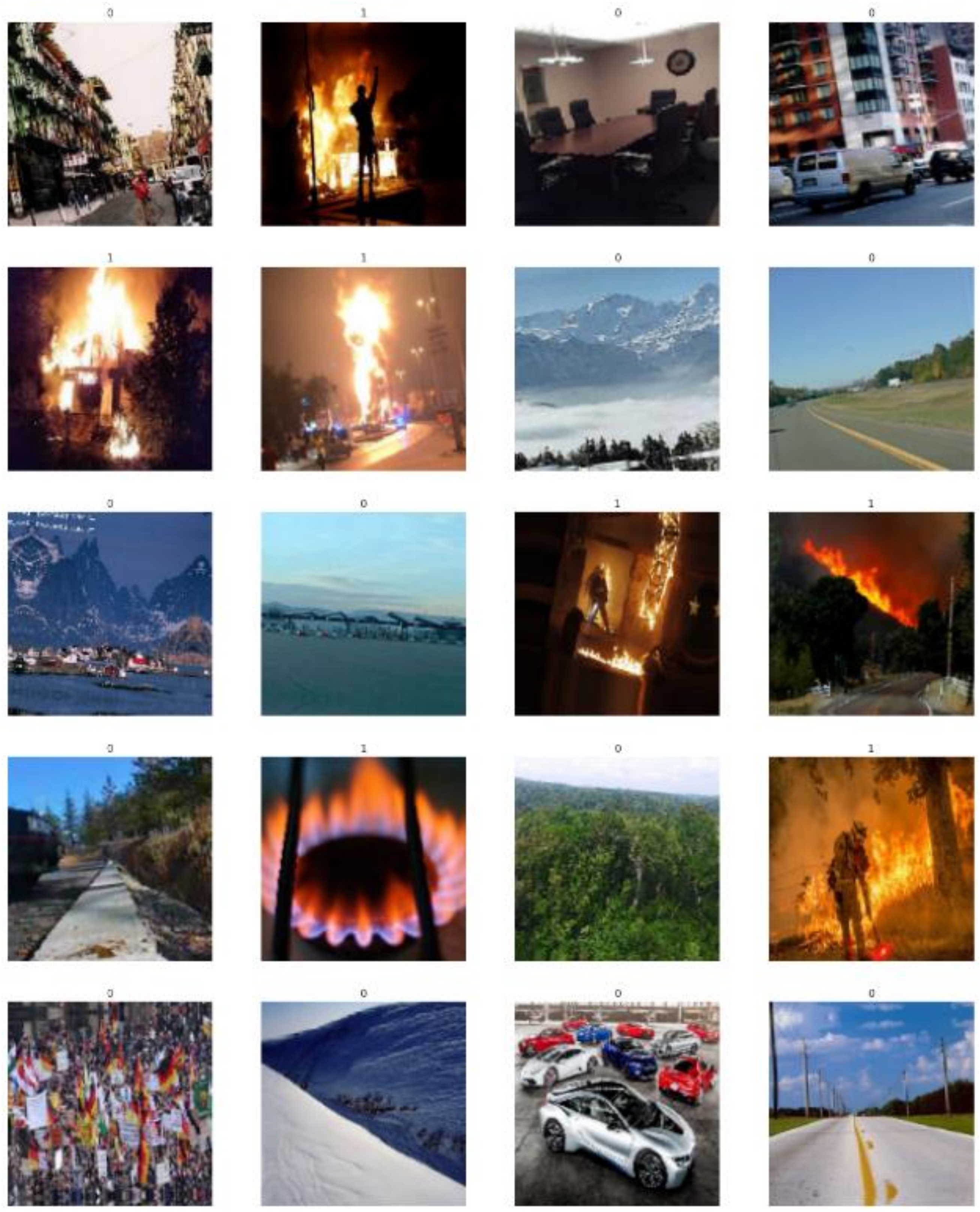


Figure 3.8 Sample-4 of labeled images from the training dataset

3.3. System Components

This thesis presents a new robust and decentralized two-stages approach for fire detection in urban areas (indoor and outdoor) through real-time data analysis for the live video stream captured by digital surveillance camera (i.e. IP camera) using image processing and state-of-the-art machine learning algorithms. The proposed algorithm contains two stages, as shown in Figure 3.9.

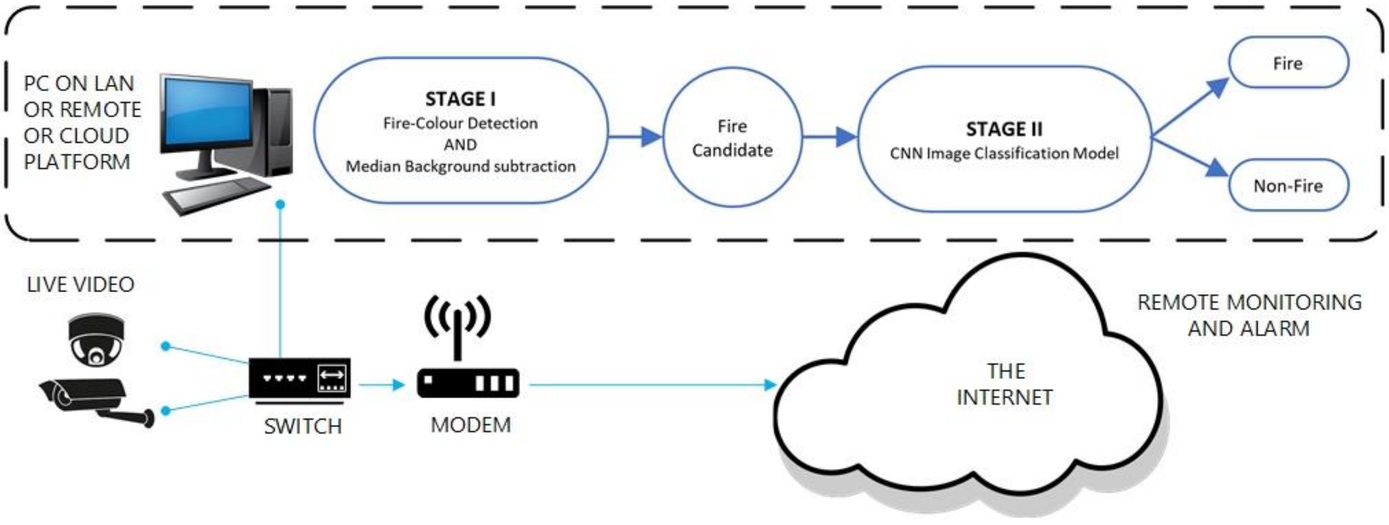


Figure 3.9 Overview of The Proposed Flame Detection System.

Stage-I: extracting the fire candidate region from the video stream based on colour and motion information; and **Stage-II**: passing the image of the fire candidate region to a Convolutional Neural Network (CNN) image classification model to classify the image as fire or non-fire.

Surveillance cameras will be everywhere in future smart cities: in homes, institutional buildings, schools, university campuses, commercial and industrial buildings, warehouses, public sites, community centers, streets, roads, to name a few. Hence, computer-vision based real-time fire detection sounds like an inexpensive yet effective approach for early fire detection and alarm indoor, in open and large spaces, as well as in outdoor applications.

Inexpensive: the proposed system does not require a high initial cost since it will run on top of the same infrastructure of the existing digital security & surveillance system network.

Effective: because of its fast response compared to traditional fire detection systems. In addition to its validity in almost all environments indoor and outdoor.

Furthermore, the system is easily **expandable** by adding more IP cameras to the existing security cameras & surveillance system network.

The proposed system structure consists of the following components, Figure 3.9:

1. Digital (IP) camera(s) to monitor the indoor/outdoor scene of interest. The cameras can be connected to the head end equipment either by wires or wirelessly;
2. Head end equipment: network switch, internet router/ modem, and local PC to store and process the video data. Otherwise, the storage and processing can be done remotely on a remote computer or on the cloud;
3. Computer program that can run on the local computer or remotely on the cloud to analyze the acquired video data in real-time and take decision.

The developed software program can run on a local computer or remotely on the cloud. This option gives the system the appeal of flexibility and versatility. In case of fire, the system will trigger local alarms and send fire alarm messages to remote destinations over the internet in the real-time. So that the system will provide early fire alarm signal for manned and even unmanned spaces.

Because of the rapid developments in digital Cameras, IoT, and 5G telecommunication technologies, computer-vision based fire detection is getting more attention from researchers and becoming a trend. Computer-vision based fire detection can be as simple as a single IoT camera serving as a complete standalone system. The camera can detect a fire early before becoming totally out of control and turning into a threatening risk, triggers a local alarm, and sends remote warning signals to the fire department and emergency management officials.

The objective of this research framework is to build a prototype real-time computer-vision based flame detection system for indoor and outdoor applications. OpenCV-Python programming environment has been chosen to implement the algorithm. OpenCV is an open-source library for computer vision algorithms that used for all sorts of image processing and video analysis in real time. In addition, the Keras library has been used to implement CNN image classification model. Keras is a high-level neural network library that runs on the top of TensorFlow, TensorFlow is a low-level open-source platform for artificial intelligence applications.

As discussed at the end of chapter 2, the reviewed computer-vision based fire detection algorithms can be broadly classified into simple computations (single-stage algorithms) and expensive computations (multi-stages algorithms). The single-stage algorithms are unreliable for urban area applications. While the multi-stages algorithms gave better results than the single-stage algorithms, however, they suffer from two main disadvantages: first, not suitable for real-time applications or require powerful hardware to run in real-time, and second, their performance deteriorates when the fire blob is small at the beginning of the fire incident or when the fire is far from the camera. The main innovation in the proposed approach is its simplicity and suitability for real-time use without compromising the accuracy and reliability in urban area applications. Thanks to the CNN image classification model that facilitates image analysis computation at the pixel level.

The proposed approach composed of two stages. **In the first stage**, colour and motion filters are applied to detect the fire-coloured moving object. The captured video stream is analyzed to extract the suspicious fire-coloured object that satisfies two conditions: first, its colour looks like fire colours ranging from different degrees of light red, orange, golden, amber, to yellow. Second, that object is moving, and it is new to the scene. Stationary fire-coloured objects such as walls, furniture, street lighting, street signboards, vehicles parked to the curb of the road etc. will be dropped off from this filtering process. It is obvious that not all fire-coloured moving objects represent a real fire, it could be people with fire-coloured clothes, or moving cars, trucks, and so on. Therefore, the proposed algorithm will automatically save the frame of the fire candidate for further processing to confirm whether it represents a real fire or not. **In the second stage,** the fire candidate image is passed to a trained CNN fire detection model to classify the image as fire or non-fire. The involvement of deep learning algorithm enhances the reliability, accuracy, and drastically reduces the number of false alarms.

The analysis shows that the proposed model is able to classify the fire images with an accuracy of 100% for the training images and 98% for the validation images. In the remaining part of this chapter, the two stages of the proposed flame detection algorithm are discussed in further detail.

3.4. Stage-I: Detecting the Fire Candidate Image in Live Video Stream

3.4.1. Foreground Detection

Foreground detection, or simply enough the frame difference [38], is an algorithm that estimates the background and performs frame-wide background subtraction to extract the foreground region for further processing. In case of fire detection problems, the dynamic of the uncontrolled fire is characterized by a large static base (i.e., the burning body) at the bottom and smaller flames that keep flickering randomly on the top of the burning body. So, most of the examined motion detection algorithms could extract only the small flickering flames and miss the large static base.

It is crucial to detect the whole body of the fire region to classify it accurately. Applying algorithms such as hybrid background estimation [35] and a mixture of Gaussian background modelling [36] causes most pixels from the base of the fire region to be modelled as background pixels. That leads to losing most of the fire region pixels and making the fire detection more challenging. After several experiments with different algorithms, the temporal median background subtraction algorithm [38] seems to give a reliable foreground model representing the whole fire region while keeping low computation cost and fast processing. Therefore, the temporal median background subtraction was found appropriate to the objective of this study, which is to use a low-cost computation that is easy to run on an average computer or IOT camera. Besides being simple in calculations, the median estimation has the advantage of eliminating the extreme pixel outliers (maxima and minima), resulting in a more noise immune foreground model.

In this technique, the background image is estimated as the median of the grayscale pixel values in the previous n frames, then the moving pixel is determined as the absolute difference between the current frame and the background model. Consider X as the ordered list of grayscale pixel values in n number of successive frames. The temporal median is given by Eq. 99, as follows:

$$Temp\_Med(X) = \begin{cases} X\left[\frac{n}{2}\right] & if\ n\ is\ even \\ \\ \frac{X\left[\frac{n-1}{2}\right]+X\left[\frac{n+1}{2}\right]}{2} & if\ n\ is\ odd \end{cases} \quad \text{……….. (99)}$$

In this study, a one-second video recorded by a 30 frames-per-second camera is used to estimate the median background. That is equivalent to 30 frames (i.e., n= 30 in Eq. 99 above). So, the first 30 frames are appended in an array then the pixel-wise median frame is calculated as given by Eq. 99 above (which represents the first estimated background). The calculated temporal median background is updated periodically every 20 seconds. This background update interval is carefully selected to be less than 30 seconds: So, it is short enough to capture a small fire before it becomes completely out of control and turns into a threatening risk [26]. Also, at the same time, the 20 seconds interval is long enough to allow for capturing the whole body of the developing fire region as a foreground (as explained above). Lastly, the moving object (the foreground mask) in the monitored scene is obtained from the absolute difference between the current frame and the temporal median background in grayscale.

3.4.2. Fire-Coloured Object Detection

Flame caused by burning common combustible materials that are found in the urban areas (e.g., wood, plastic, paper, etc.) is distinguishable by high brightness, high contrast to surrounding pixels, and the reddish-yellow and amber colours. Those colour and brightness features are used to extract the fire-coloured region in the frame. YCbCr, YUV, and HSV colour spaces are commonly used in the literature to detect and extract colour regions. That is because those colour spaces separate the illumination information (luma) from the colour information (chroma) [4], [8], which makes the colour extraction model much easier to implement and more robust to changes in illumination and weather conditions.

The YCbCr colour space has an additional advantage: the relation between the red-difference chroma (Cr) and the blue-difference chroma (Cb) is mutually exclusive, making it ideal for testing the fire-coloured pixels precisely. The same colour extraction rules described in [4] (in YCbCr colour space) are used in this proposed algorithm. Similar colour rules are used in [8] but in the YUV colour space. So, the following colour rules have to be all true to classify the pixel as fire-coloured pixel. The colour rules are given by Eq. 100 thru Eq. 103, as follows:

Rule 1: $Y(m,n) > Cb(m,n)$ ………………………………………………….. (100)

Rule 2: $Cr(m,n) > Cb(m,n)$ …………………………………….…...….. (101)

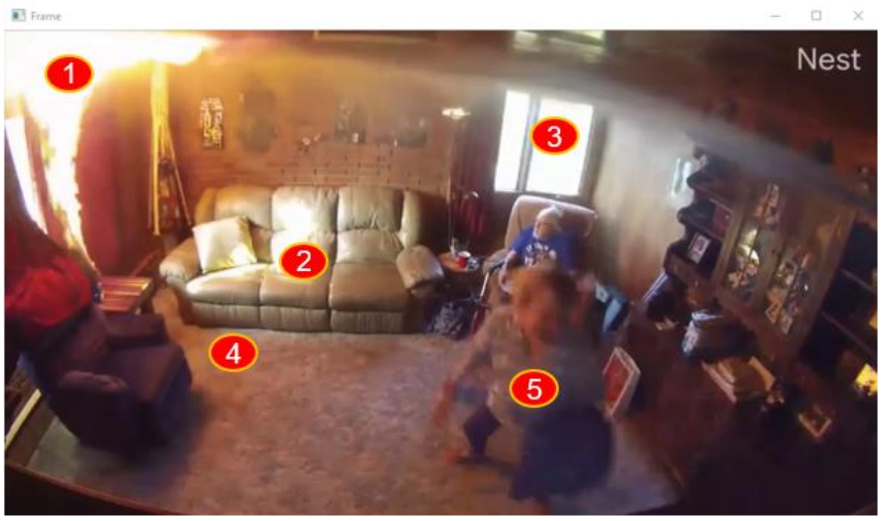
Rule 3: $Y(m,n) > Y_{mean} \cap Cr(m,n) > Cr_{mean} \cap Cb(m,n) < Cb_{mean}$ …….. (102)

Rule 4: $|Cr(m,n) - Cb(m,n)| \geq \tau \qquad , \ \tau = 40$ ………………….……... (103)

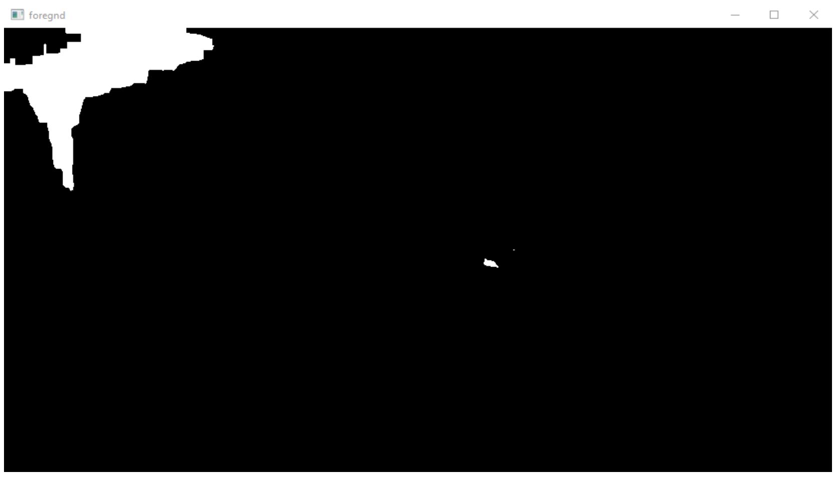
Where Y is the luma component, Cr is the red-difference chroma component, and Cb is the blue-difference chroma component. Y_{mean} , Cr_{mean}, and Cb_{mean} are the frame-wise mean components. Finally, two separate masks have been extracted from the original frame: one represents the moving object (the foreground mask), and the other represents the fire-coloured region (the fire-coloured object mask). Then, bit-wise logic 'AND' operation is applied to merge both masks into a single mask representing the fire-coloured AND moving objects in the frame, which will be referred to as fire candidate from now on.

3.4.3. Testing the Fire Candidate Detection Algorithm

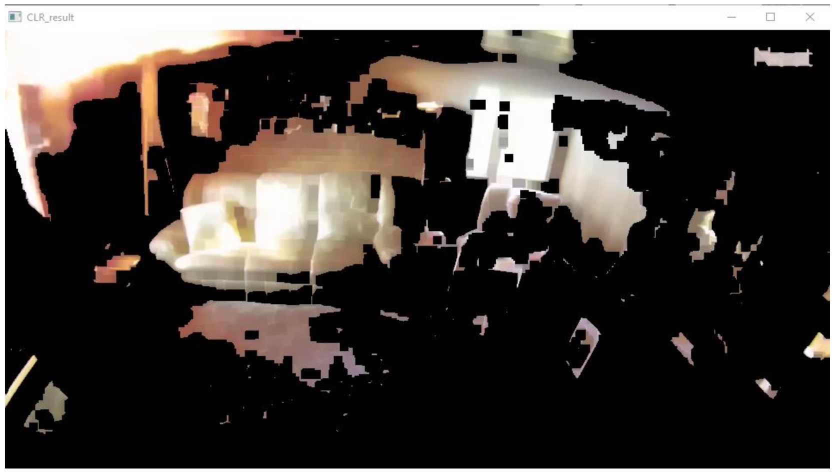
Sample video clips have been used to test the performance of the fire candidate detection algorithm. Figure 3.10 shows one frame extracted from a sample video clip [2]. The original frame shows several fire-coloured objects: 1) the fire on the left-hand side, 2) the sofa, 3) the window, and 4) part of the walls and ground. The frame also shows 5) speedy motion of a panic lady. Applying the fire candidate detection algorithm on this clip resulted in only the fire-coloured moving object to be detected (which is a real fire in this video). While the stationery fire-coloured objects (the window, the sofa, the walls, and the ground) have been omitted from the final result. Similarly, the motion of the lady has been omitted from the final results thanks to the colour filtering rules. Applying the algorithm to other sample video clips gave the same results. The conclusion is the fire candidate detection algorithm works as intended to extract only the fire-coloured moving objects.



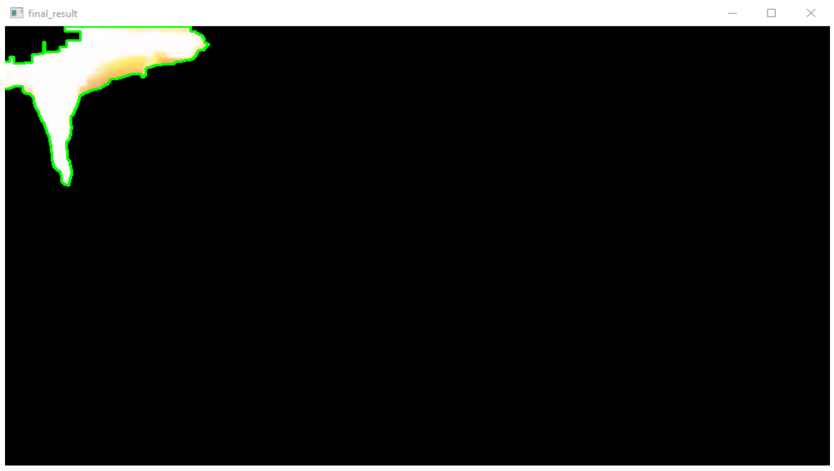
(a) Original Frame. [2]



(b) Foreground Mask.



(c) Colour Mask.



(d) The Fire Candidate (fire-coloured moving object).

Figure 3.10 Fire candidate Detection.

3.4.4. Comments on Stage-I Results

The fire candidate may represent real fire (as in the previous sample video) or just a fire-coloured moving object. Examples of fire-coloured moving objects are people in fire-coloured clothes, fire-coloured vehicles, flags, signs, and so on so forth.

Figure 3.11 shows sample photos of fire-coloured moving objects found in the urban areas. Therefore, the colour and motion information only are not enough to decide whether the fire candidate represents a real fire or not. In conclusion, a second image classification stage is required to classify the obtained image as fire or non-fire based on the salient features of the fire candidate.

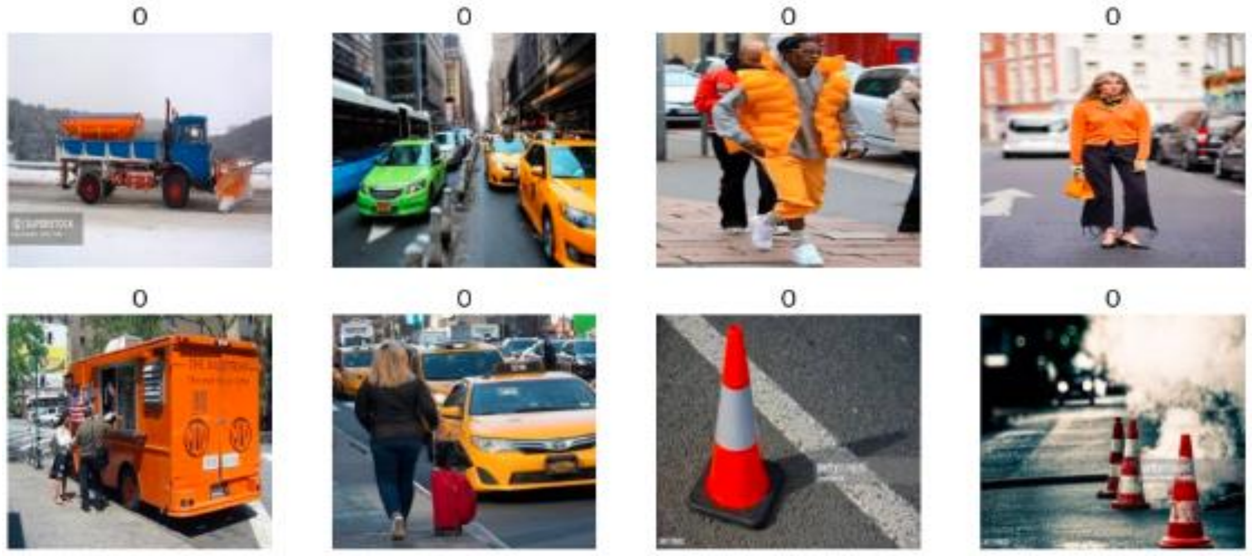


Figure 3.11 Examples of fire-coloured objects found in upran areas.

The fire candidate size and continuity have to be tested before moving to the second stage. If the size of the fire candidate is larger than a threshold percentage of the frame size (5% of the frame size), then a conditional frame counter is started to count the number of successive frames containing that fire candidate. If the fire candidate continues for 90 consecutive frames (3 seconds in case of a standard 30 frames per second camera), the last frame is saved and passed to the second stage. Checking the size and continuity of the fire candidate is an essential step to reduce the effect of noise and minimize the number of false alarms. In the second stage, the fire candidate image is processed by a trained CNN image classification algorithm to classify it as real fire or non-fire.

3.5. Stage-II: CNN Image Classification Algorithm

In the previous stage, the colour and motion information are used to extract the fire candidate. It has been proven that the colour and motion information alone are not enough to decide whether the fire candidate represents a real fire or non-fire (binary image classification problem). Therefore, the second stage of testing is required to improve efficiency and reduce false alarms. Common Machine Learning (ML) image classification algorithms used in the literature have been explored, looking for the best solution for that binary classification problem. It is found that, for successful classification of a small fire blob at an early stage, the value of each pixel in the image and its relation to the surrounding pixels have to be considered in the feature extraction computations. Therefore, the deep learning (DL) Convolution Neural Network (CNN) model is proposed to automate the process of feature extraction and image classification.

3.5.1. The Proposed Architecture

Since the problem is defined to be a binary image classification problem, the model is built with the least possible number of layers. A deeper model will be expensive in computations and most probably will lead to overfitting. On the other hand, a smaller model will give an acceptable generalized solution without compromising accuracy. Eventually, the proposed model size after training is only 9.58MB which makes it more suitable for a real-time fire detection system that requires low-power hardware to run it. The proposed model (Figure 3.12) consists of three Convolution layers, each is followed by a ReLU activation and Max pooling layers, pool size (2X2). All the Conv layers use small receptive field sizes (3×3) and dilated Convolutions with a dilation rate of two. Zero padding is used to keep the image size unchanged and to maintain the corner pixel contribution to the extracted features.

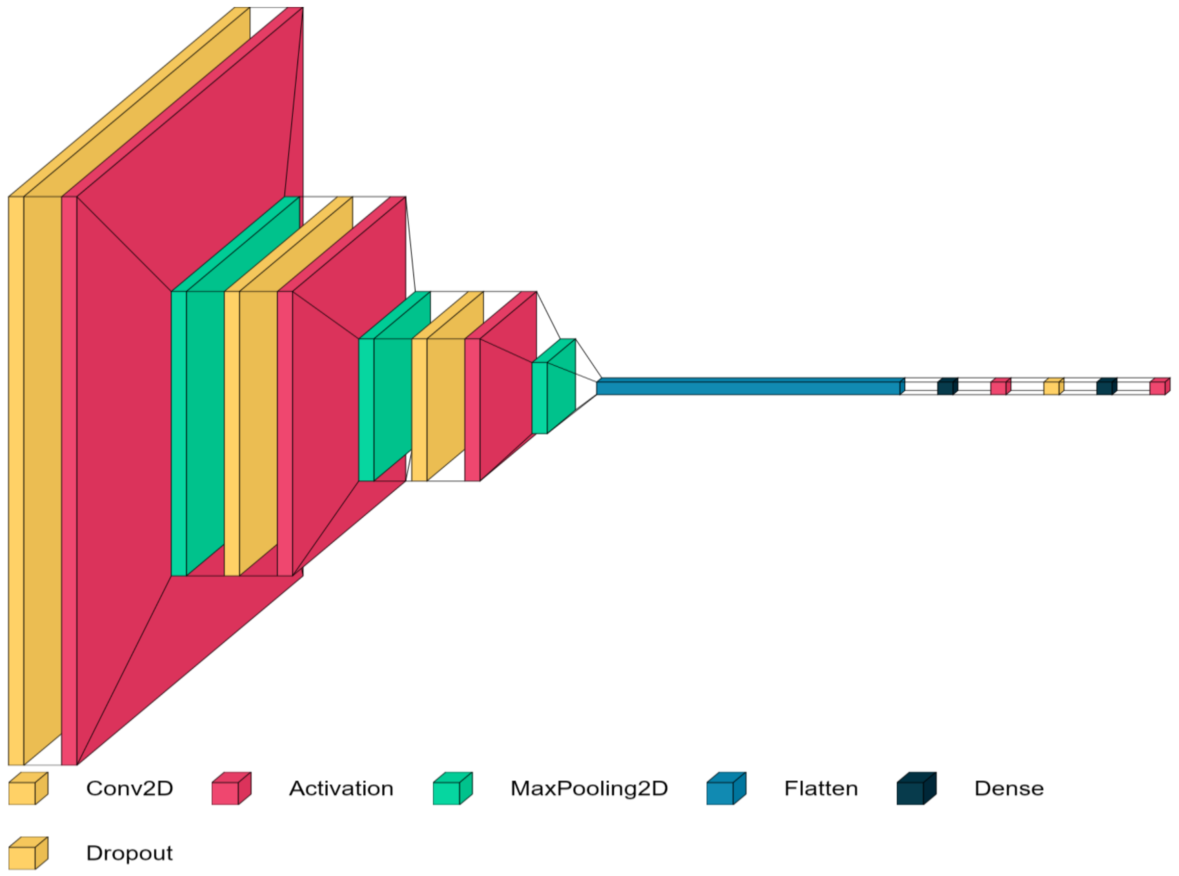


Figure 3.12 3D visualization for the architecture of the proposed CNN model.

Then the output of the third Conv layer is flattened and connected to the first dense (fully connected) layer, and another ReLU activation, prior to dropping out 50% of the neurons and passing to the final dense (fully connected) layer and then Sigmoid activation function with two output neurons represent the binary output: fire/non-fire. The proposed CNN Model flow diagram is shown in Figure 3.13.

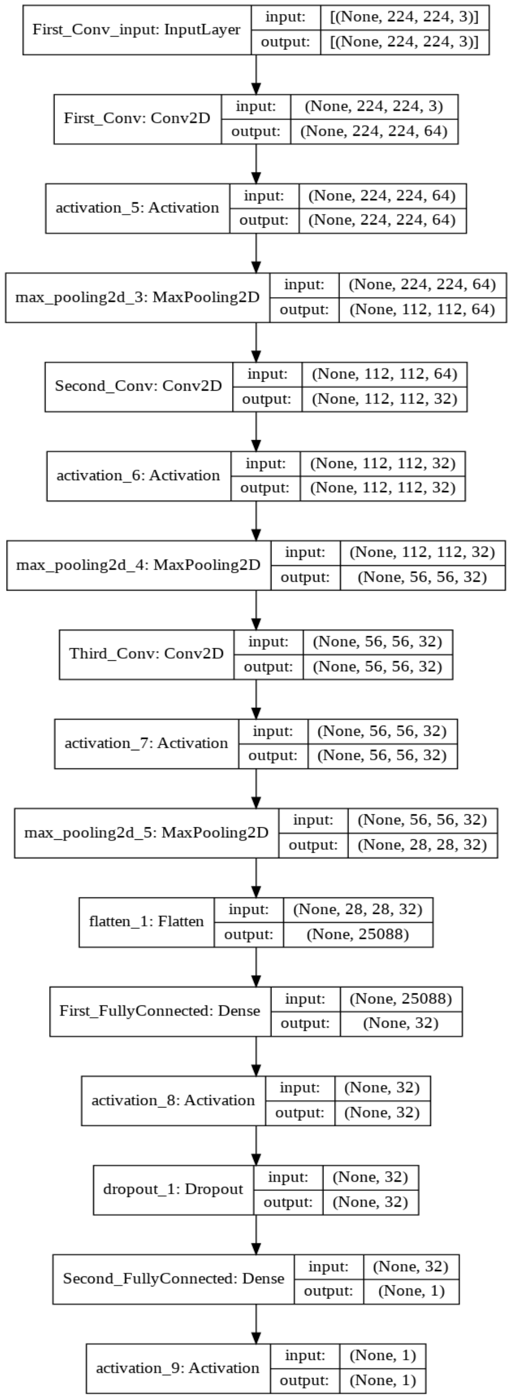

First_Conv_input: InputLayer
input: [(None, 224, 224, 3)]
output: [(None, 224, 224, 3)]
First_Conv: Conv2D
input: (None, 224, 224, 3)
output: (None, 224, 224, 64)
activation_5: Activation
input: (None, 224, 224, 64)
output: (None, 224, 224, 64)
max_pooling2d_3: MaxPooling2D
input: (None, 224, 224, 64)
output: (None, 112, 112, 64)
Second_Conv: Conv2D
input: (None, 112, 112, 64)
output: (None, 112, 112, 32)
activation_6: Activation
input: (None, 112, 112, 32)
output: (None, 112, 112, 32)
max_pooling2d_4: MaxPooling2D
input: (None, 112, 112, 32)
output: (None, 56, 56, 32)
Third_Conv: Conv2D
input: (None, 56, 56, 32)
output: (None, 56, 56, 32)
activation_7: Activation
input: (None, 56, 56, 32)
output: (None, 56, 56, 32)
max_pooling2d_5: MaxPooling2D
input: (None, 56, 56, 32)
output: (None, 28, 28, 32)
flatten_1: Flatten
input: (None, 28, 28, 32)
output: (None, 25088)
First_FullyConnected: Dense
input: (None, 25088)
output: (None, 32)
activation_8: Activation
input: (None, 32)
output: (None, 32)
dropout_1: Dropout
input: (None, 32)
output: (None, 32)
Second_FullyConnected: Dense
input: (None, 32)
output: (None, 1)
activation_9: Activation
input: (None, 1)
output: (None, 1)

Figure 3.13 The proposed CNN Model flow diagram.

3.6. Summary

This chapter starts with demonstration of the efforts made to collect and prepare the image and video datasets required to train, test, and validate each stage of the proposed algorithm. The concept of data augmentation has been discussed. Next, an overview of the proposed system and the main system components have been presented. The proposed two-stages early fire detection algorithm has been discussed in detail. The proposed flame detection system involves two stages: Stage-I: extracting the fire candidate region from the live video stream in real-time using colour and motion information, and Stage-II**:** passing the fire candidate region to a pre-trained Convolutional Neural Network (CNN) model to classify the image as fire or non-fire. In-depth demonstration of the design of the proposed CNN image classification model has been provided at the end of the chapter.

Chapter 4. CNN Model Implementation, Performance, and Metrics

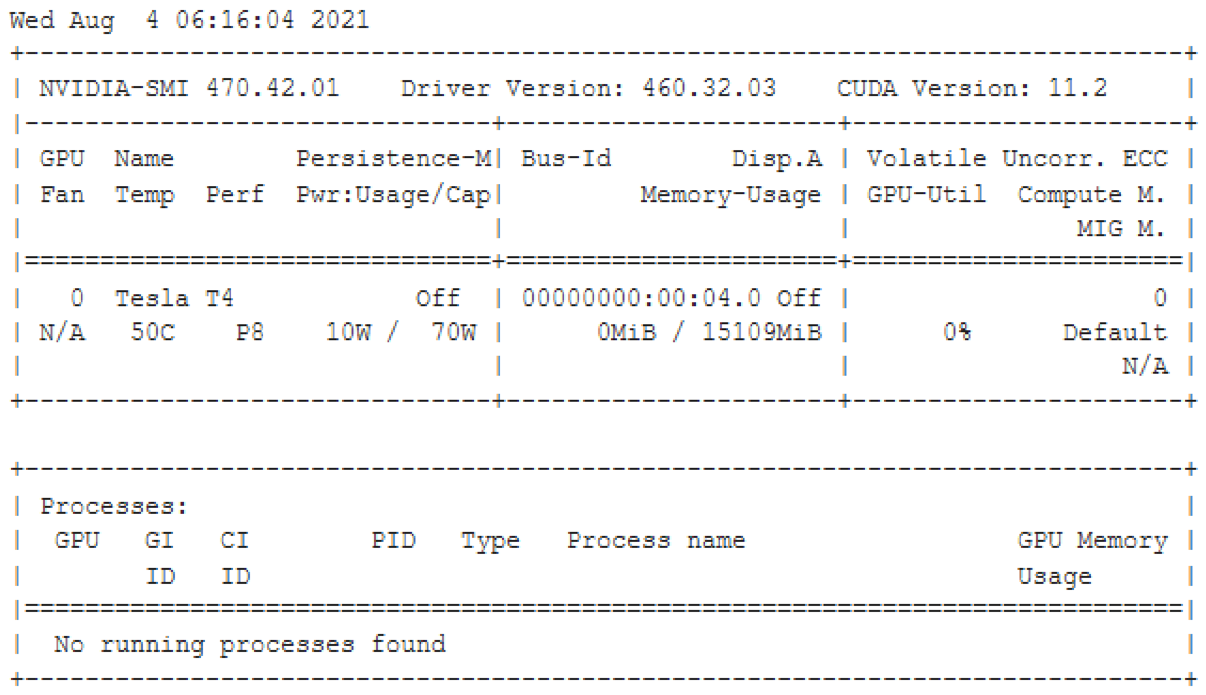
4.1. Introduction

This chapter first presents the implementation of the proposed CNN model. The model optimization attempts are discussed after. Then, the effects of the learning rate on the model stability and dynamics are demonstrated. Next, the concept of adaptive learning rate is presented. Later, the Adaptive Moment (Adam) optimization algorithm and its advantages are presented. The chapter ends by reviewing the model performance and results; five performance metrics are presented: confusion matrix, accuracy, precision, recall rate, and F-score.

4.2. Implementation

The Keras library is used to implement the proposed CNN model; Keras is a well-known high-level neural network library that runs on the top of TensorFlow. TensorFlow is a low-level open-source machine learning platform. The modules and algorithms of the Keras library are built-in Python language, which makes it easier to learn and apply than TensorFlow.

Google Colab programming platform is used to build and run the code. Google Colab (https://colab.research.google.com/) is a product of Google Research. Google Colab allows the user to write and execute python code through the internet browser, and it is especially well suited to machine learning, data analysis and education. Furthermore, Google Colab provides access to powerful computing resources, including GPUs (graphics processing units) on the cloud, for faster model training. In this experiment, Google Colab provides a fast GPU (model Tesla T4 Nvidia, Figure 4.1) which enabled the model training using 4300 images in around 106 seconds per epoch (attempt). This time includes the data augmentation process, as explained earlier. A standard CPU model Intel(R) Core i7-3720QM @ 2.60GHz with 16 GB RAM (and no dedicated GPU) took approximately 600 seconds per epoch to train the same model with the same dataset.



```
Wed Aug  4 06:16:04 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.42.01    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   50C    P8    10W /  70W |      0MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

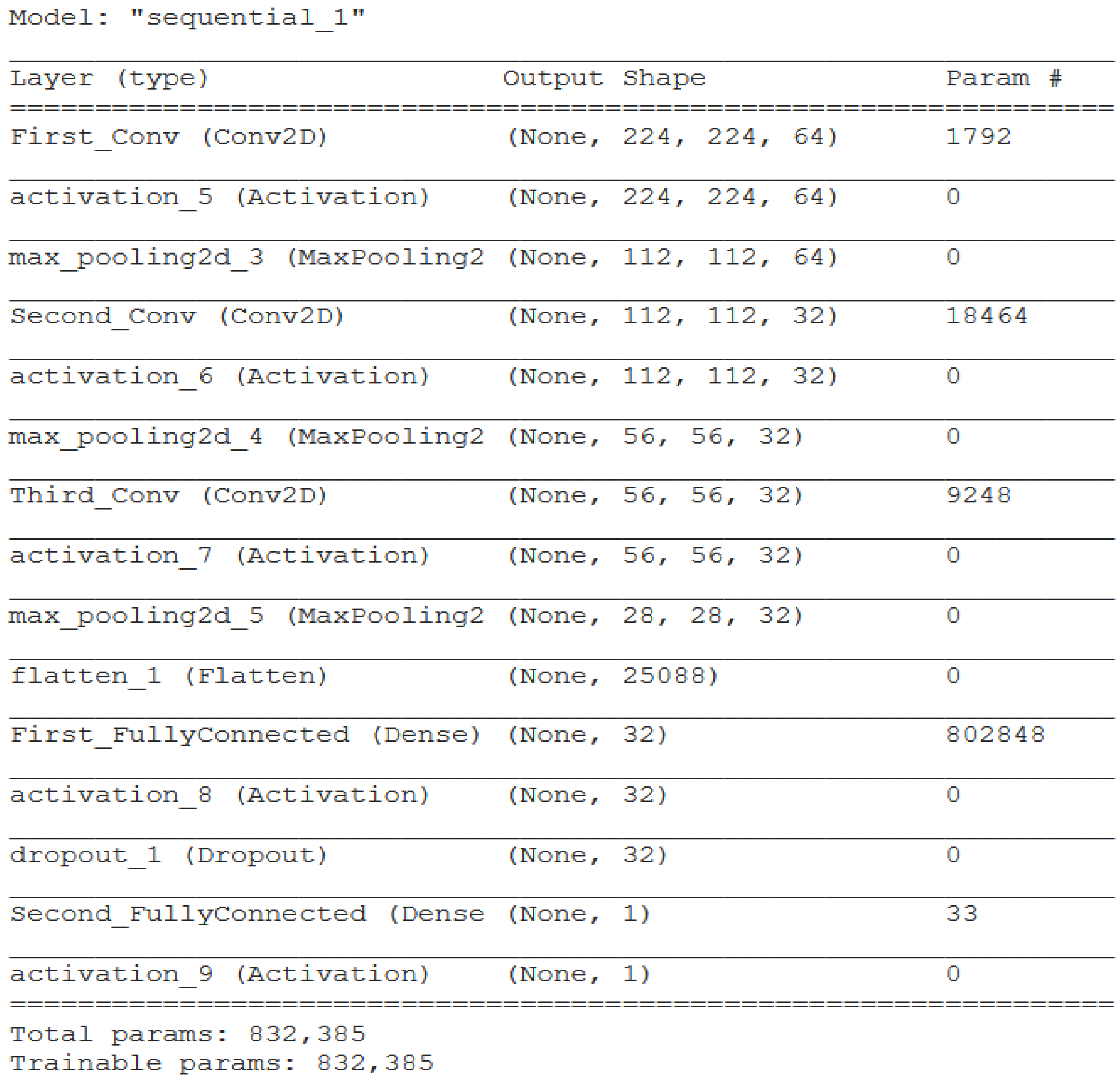
Figure 4.1 Specifcation of the GPU used to train and test the model.

4.3. Model Optimization

In order to build a simple yet efficient model, the model architecture starts with the most straightforward configuration: including an input and Conv layer followed by a flattening and dense layer, then the binary output layer. Later, additional hidden Conv layers have been added to the model, one at a time to add more non-linearity to the calculations, while watching the improvement in the model performance measured by the accuracy of the image classification for both the training and the validation datasets. Max pooling and 50% dropout layers are used for two equally vital purposes: first, to simplify the computations by reducing the number of the learnable parameters (i.e., the weights and biases of the CNN) hence reducing the model complexity, and second, to eliminate overfitting. Besides, the Max pooling layer reduces the size of the processed image; so that more global features are extracted for the same kernel size.

Next, several trial-and-error attempts have been carried out to improve the model performance: by selecting the appropriate size of the input images, the optimum hyperparameters such as the number of filters (aka kernels) in each Conv layer and their size, the dilation rate, the number of epochs, the batch size, and choosing the initial value of the learning rate and its optimizer. The proposed CNN model summary is shown in Figure 4.2, and the final values of the CNN model hyperparameters are listed below:

- Input image size is 224 X 224 X 3 (RGB coloured image);
- The total number of epochs is 200 (one epoch means training the neural network model with all the training images for one cycle forward and backward);
- Batch size is 32 (this is the number of images that processed by the model simultaneously in a single batch);
- Learning rate optimizer is Adam (this will be discussed in detail in the next sections).



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
First_Conv (Conv2D)          (None, 224, 224, 64)      1792
_________________________________________________________________
activation_5 (Activation)    (None, 224, 224, 64)      0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 112, 112, 64)      0
_________________________________________________________________
Second_Conv (Conv2D)         (None, 112, 112, 32)      18464
_________________________________________________________________
activation_6 (Activation)    (None, 112, 112, 32)      0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
Third_Conv (Conv2D)          (None, 56, 56, 32)        9248
_________________________________________________________________
activation_7 (Activation)    (None, 56, 56, 32)        0
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 28, 28, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
First_FullyConnected (Dense) (None, 32)                802848
_________________________________________________________________
activation_8 (Activation)    (None, 32)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
Second_FullyConnected (Dense (None, 1)                 33
_________________________________________________________________
activation_9 (Activation)    (None, 1)                 0
=================================================================
Total params: 832,385
Trainable params: 832,385
```

Figure 4.2 The proposed CNN Model summary.

4.4. Dynamics of The Learning Rate and The CNN Model Behavior

During the training process, the optimum weights of the neural network model are updated empirically via a procedure called stochastic gradient descent by using samples from the training dataset. That process is controlled by the learning rate.

The learning rate is the most crucial hyper-parameter that controls how much to change the deep learning neural network model in response to the calculated error (loss) between the predictions and the ground truth at each time the model weights are updated. The contribution of each weight to the total loss (i.e., the gradient of the error with respect to each weight) is calculated through the backpropagation path. Then, instead of updating the weights with the full amount, it is scaled down by the learning rate (typically between 0 and 1). As a metaphor, the learning rate is the size of the step towards the optimum solution where the error (loss or cost) is minimum, Figure 4.3. Usually, the learning rate is relatively high (between 0.01 and 0.001) at the beginning of training, then the rate decreases as the training progress (fine-tuning).

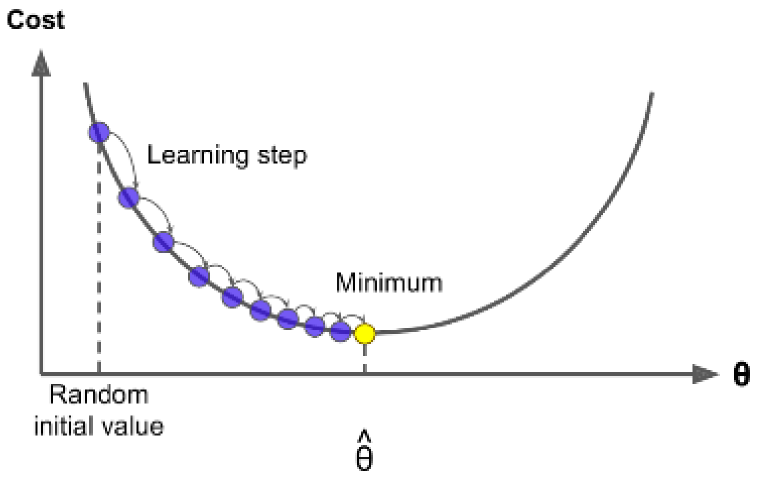


Figure 4.3 DL CNN model learning.

Finding the proper learning rate is a challenging task. Too small learning rate leads to slow learning process and increases the number of epochs required to achieve acceptable accuracy. Or even the worst, the model may be stuck to a local minimum and stop improving (i.e. failure to train). On the other hand, a high learning rate increases the possibility of overshooting and missing the global minimum. Or even the worst, it may lead to model divergence and instability, Figure 4.4.

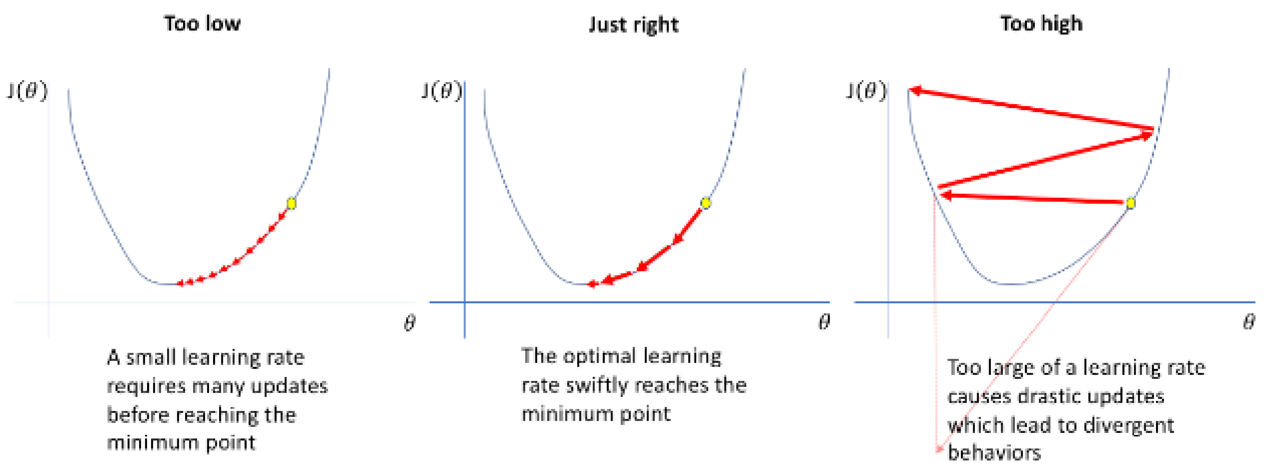

Too low
J(θ)
θ
A small learning rate requires many updates before reaching the minimum point
Just right
J(θ)
θ
The optimal learning rate swiftly reaches the minimum point
Too high
J(θ)
θ
Too large of a learning rate causes drastic updates which lead to divergent behaviors

Figure 4.4 Effect of learning rate on the DL NN model performance.

4.5. Learning Rate and Adaptive Optimization

It is not possible, analytically, to calculate the optimal learning rate or the learning rate decay schedule. Such values are discovered by trial and error. Alternatively, the learning algorithm can monitor the model performance on the training dataset and adjust the learning rate accordingly. This technique is known as the adaptive learning rate. Adaptive learning rate speeds up the training process while eliminating the hassle of choosing a learning rate and its decay schedule [14].

Adam optimizer [14] is one of the popular adaptive learning rate optimization methods. It is built on stochastic gradient descent. Adam optimizer is proven to be robust over different NN architectures and various types of problems. The name ‘Adam’ is derived from Adaptive Moment (or Momentum) estimation. Adam optimizer initializes the learning rate and adapts its value as the training progresses, one learning rate per model weight. The initial values of the Adam optimizer are shown in Figure 4.5. The training of the proposed CNN model uses Adam optimizer.

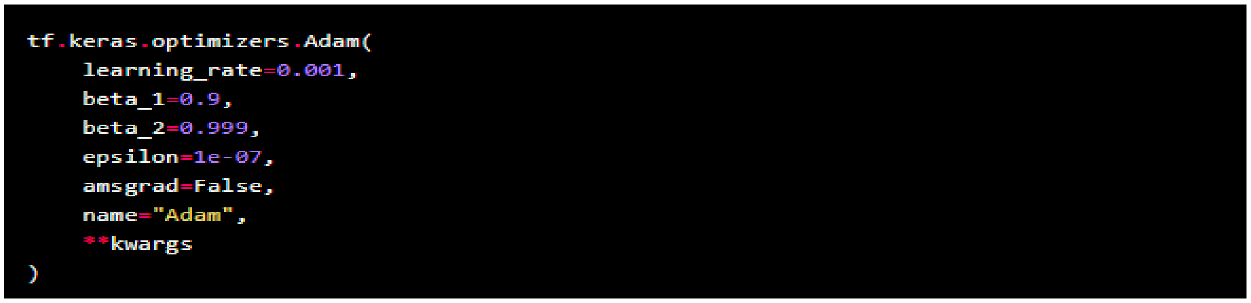

```
tf.keras.optimizers.Adam(
    learning_rate=0.001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-07,
    amsgrad=False,
    name="Adam",
    **kwargs
)
```

Figure 4.5 Implementation of Adam algorithm on Keras – TensorFlow.

According to the authors [14], Adam optimization algorithm has the following advantages:

- Computationally efficient
- Use less memory compared to other optimization algorithms
- Well suited for large data problems as well as problems with large number of parameters
- The hyper-parameters have intuitive interpretation and require little to no tuning.
- Gradient descent algorithm uses constant learning rate, whereas Adam computes adaptive learning rates.

4.6. The Results and Performance Metrics

Following the optimization techniques mentioned above, the validation accuracy started to improve quite fast, from around 75% in the beginning to +80%, +85%, then +90% and eventually 98%. At this stage, the training data accuracy reached 100%. It is proved in this study that using a small number of layers and a small filter size in each layer can improve the performance and generalizability of the fire/non-fire image classification thanks to the dilated convolution. This approach could overcome the problem of overfitting for limited datasets. In the following sub-sections, the model performance metrics are presented. Explanation and equations of those metrics are reviewed in chapter 2, sub-sections 2.4.5 and 2.4.6.

4.6.1. Confusion Matrix

Confusion Matrix provides a detailed overview on the results of the binary classification model.

The results of the proposed CNN image classification model are given follows Figure 4.6):

- True Positive (TP) = 483,
- False Positive (FP) = 12,
- True Negative (TN) = 486, and
- False Negative (FN) = 11

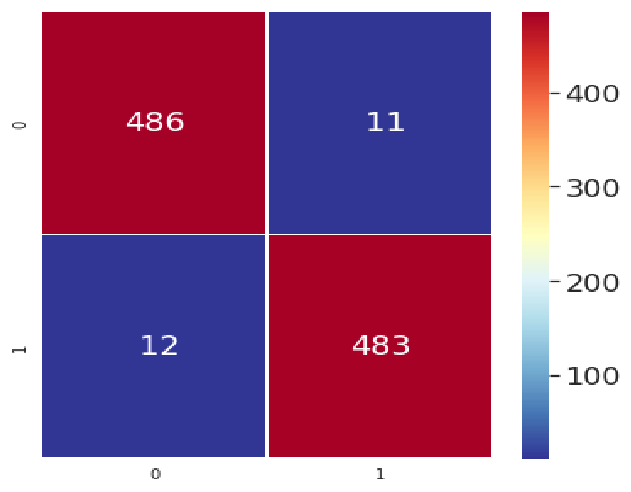


Figure 4.6 The Confusion Matrix.

4.6.2. Accuracy

Accuracy [9] is the proportion of the correct classifications among the total number of classifications. Training and validation accuracy curves are shown in Figure 4.7. The final values of the model accuracies after 200 epochs are shown in Figure 4.8

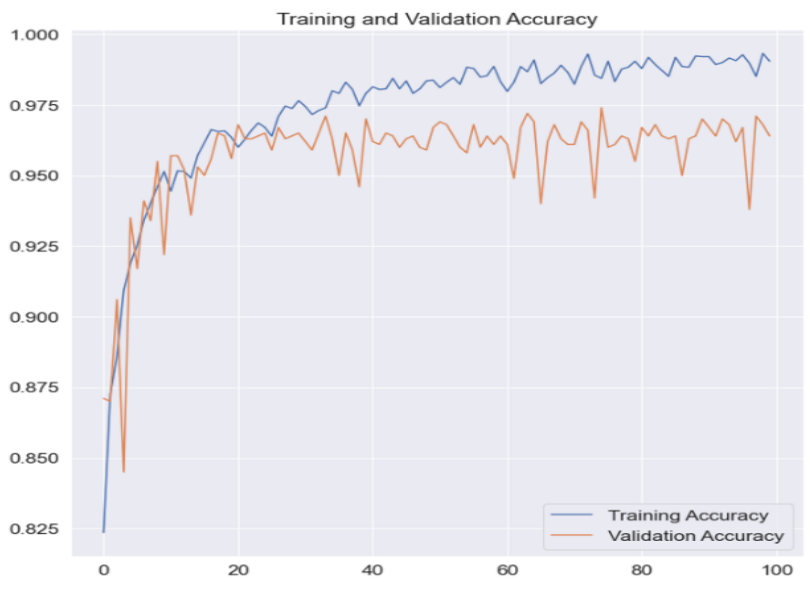
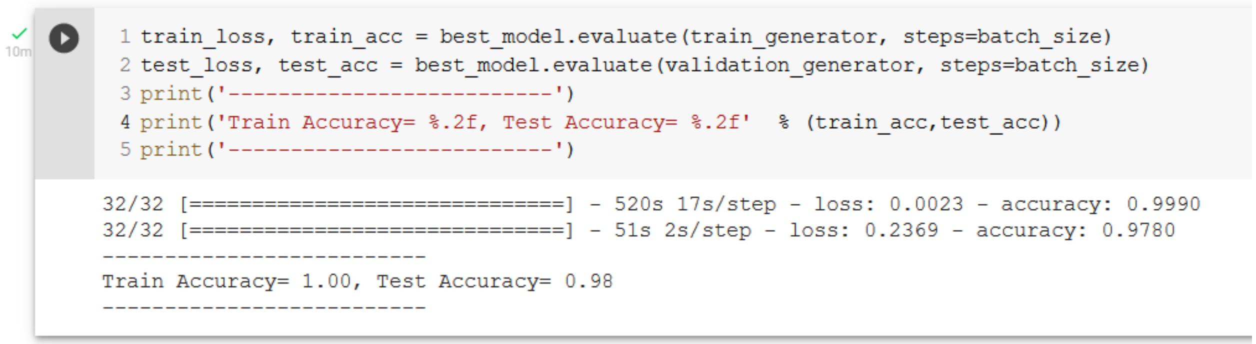


Figure 4.7 Traning and validation accuracy curves during the first 100 epochs



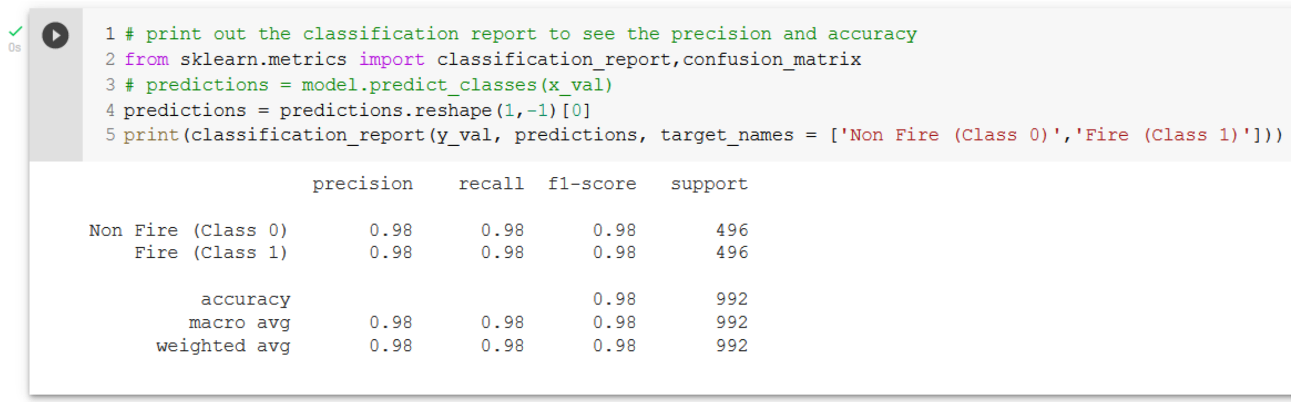
```
train_loss, train_acc = best_model.evaluate(train_generator, steps=batch_size)
test_loss, test_acc = best_model.evaluate(validation_generator, steps=batch_size)
print('--------------------------')
print('Train Accuracy= %.2f, Test Accuracy= %.2f'  % (train_acc,test_acc))
print('--------------------------')
```

```
32/32 [==============================] - 520s 17s/step - loss: 0.0023 - accuracy: 0.9990
32/32 [==============================] - 51s 2s/step - loss: 0.2369 - accuracy: 0.9780
--------------------------
Train Accuracy= 1.00, Test Accuracy= 0.98
--------------------------
```

Figure 4.8 Final traning and validation accuracy values after 200 epochs.

4.6.3. Precision, Recall, and F-Score

Precision is the proportion of the true fire detected by the model to the total fire (true or false) detected by the model. Recall (also called Sensitivity) is the proportion of the true fire detected by the model to the total number of fire instances in the dataset (detected or not). F- Score (F- Factor) provides a single number that combines both the precision and the recall. Precision, Recall, and F-Score values are shown in Figure 4.9



```
# print out the classification report to see the precision and accuracy
from sklearn.metrics import classification_report,confusion_matrix
# predictions = model.predict_classes(x_val)
predictions = predictions.reshape(1,-1)[0]
print(classification_report(y_val, predictions, target_names = ['Non Fire (Class 0)','Fire (Class 1)']))
```

```
                    precision    recall  f1-score   support

Non Fire (Class 0)       0.98      0.98      0.98       496
    Fire (Class 1)       0.98      0.98      0.98       496

          accuracy                           0.98       992
         macro avg       0.98      0.98      0.98       992
      weighted avg       0.98      0.98      0.98       992
```

Figure 4.9 Precision, Recall and F1-score values for the propsoed model after 200 epochs

4.6.4. Validation of the proposed 2-Stage approach

It is clear from the training and validation accuracy curves and the other performance metrics that the model is converging. However, there are apparent fluctuations. The fluctuation occurs because there is no single best space of solutions (sets of weights) that can conclusively differentiate between the two classes (fire versus non-fire). In other words, there is no single global minimum. Instead, there are many good solutions (global optima).

The volatile accuracy can be explained by looking at the diversified dataset used to train and validate the CNN classification model. The dataset contains many images containing fire-coloured bright objects. Classifying those images is a real challenge. For example, the below image shows a street wall sconce illuminated by the old-fashion sodium vapour electric bulb, Figure 4.10. The yellowish light emitted from the sconce and its reflection off the old building fascia creates shades of reds, yellows and oranges, which, in turn, forms a "blaze" like effect. As indicated, this image is misclassified by the CNN model as "a fire" image.

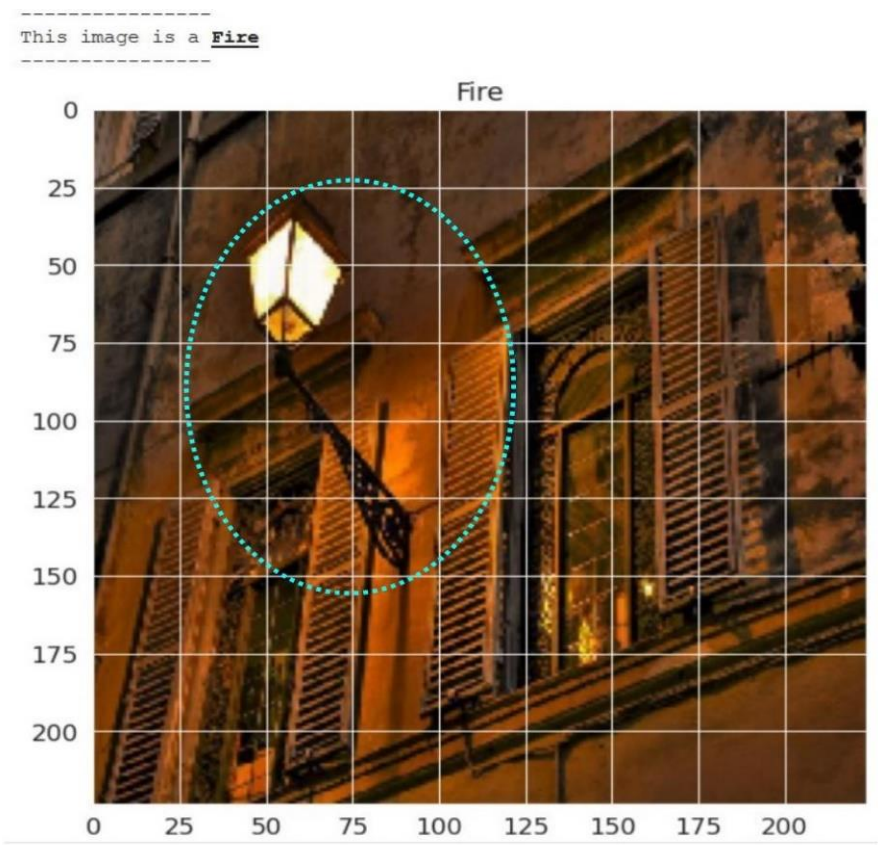

```
----------------
This image is a Fire
----------------
```

Figure 4.10 Missclassified non-fire image

The CNN classification model will randomly misclassify those scenes include bright fire-coloured objects such as street lighting, indoor lighting, and etc. Here, the importance of the first stage of the proposed model comes into the picture. Since it is a stationary object, this sconce and similar street lighting and indoor lighting will be classified as background objects in the video processing phase (Stage 1), and hence it will be filtered out and not passed to the CNN model (Stage 2). This proposed approach gives more robust performance and eliminates the misclassification possibilities caused by such tricky objects.

4.7. Summary

This chapter first presented the implementation of the proposed CNN image classification model. Model optimization attempts are discussed after. Then, the effects of learning rate on the model convergence and dynamics are demonstrated. Next the concept of adaptive learning rate is presented. The chapter ends by presenting the results and measuring the model performance through five performance metrics. The model training and validation accuracies reached 100% and 98% respectively. It is proved in this study that using a small number of layers and a small filter size in each layer can improve the performance and generalizability of the fire/non-fire image classification model. This approach could overcome the problem of overfitting for limited datasets.

Chapter 5. Conclusions and Future Research

5.1. Conclusions

The work presented in the thesis aims to develop a prototype real-time computer-vision based early fire detection system. As the literature review revealed, there are many recent methods for fire detection based on computer-vision. Those methods have several advantages, but they also still have limitations. The proposed system involves two stages: **in the first stage**, the video captured by surveillance camera is processed in real time searching for fire candidate region. The fire candidate region is identified by two strong features of fire, a) fire colours and b) motion. The first stage is implemented (coded) using Python OpenCV open-source computer programming language for computer-vision applications. **In the second stage**, the extracted fire candidate is examined by a pre-trained CNN image classification model to classify the image as fire or non-fire. The CNN model is built using deep learning library called Keras, which is a high level library built on the top TensorFlow open source platform.

Stage-I, the fire candidate detection algorithm as follows:

a. Extract the foreground moving objects from the fixed background scene using dynamic median background subtraction.
b. Extract the fire-coloured object in the frame using colour detection rules in YCbCr colour space. The YCbCr colour space is used in fire detection algorithms for two reasons: first, it separates the luminance component from the colour components efficiently more than any other colour model. Second, the relation between the red-difference chroma (Cr) and the blue-difference chroma (Cb) is mutually exclusive, which makes it ideal for testing the fire-coloured pixels precisely.
c. Perform a bit-wise logic ‘AND’ operation on the above two outputs in order to obtain a blob that is characterized by being moving AND fire-like coloured. The output of this step is the fire candidate.

The fire candidate detection algorithm has been tested using several sample videos with and without fire. The performance is always robust, and the results show only the right fire candidate characterized by two features: the fire colours and the motion.

In the second stage, the extracted fire candidate is tested for size and continuity. If the fire candidate reached at least 5% of the frame size and continues for 100 consecutive frames (around

3.3 seconds for 30 frames per second camera) then the fire candidate image is processed by a pre-trained CNN image classification model to classify the image as fire or non-fire. The CNN model is trained and tested on Google Colab platform (colab.research.google.com). Google Colab allows user to write and execute python machine learning code through the browser. Moreover, Colab provides free access to computing resources, including cloud GPUs (graphics processing units), for faster model training.

4300 real fire and non-fire labelled images have been used to train the CNN model, and 1000 labelled images have been used for validation. To further challenge the classification model, most of the non-fire images include fire-like coloured objects, for example, people with fire-like coloured clothes, fire-like coloured vehicles and flags, etc. The size of the proposed CNN model after training is only 9.58MB which makes it more suitable for a real-time fire detection system that requires low-power hardware to run it. The results shows that the proposed system is capable to classify the training and validation images with 100% and 98% accuracies respectively.

5.2. Key Contributions

This research work proposes an improved computer-vision based fire detection system for indoor and outdoor applications. The scientific contributions of this work are as follows:

Collecting the visual features of the flame region in digital images that are recognizable by image processing algorithms from the recent research works.

Development of Novel Real-Time Early Fire Detection Model: This research develops a fire detection model consisting of two stages: the first stage implies the extraction of fire candidate region from video stream captured by a surveillant camera in real-time. In the second stage, the extracted fire candidate region is tested by a trained CNN image classification algorithm to classify the region as fire or non-fire. The main innovation in the proposed approach is its simplicity and suitability for real-time use without compromising the accuracy and reliability in urban area applications. Thanks to the CNN image classification model that facilitates image analysis computation at the pixel level and carries the complex calculations in the offline training mode.

Coding of the Prototype Video Processing Model: using OpenCV-Python programming environment to implement the proposed model.

Coding of the Prototype CNN image Classification Model: using Keras /TensorFlow deep learning library to implement the CNN image classification model.

Collecting and preparing dataset of 5300 real life images for model training and validation: the images represent real fire and non-fire scenes from the urban community, indoor (in buildings and houses), as well as outdoor (in streets, roads, and landscapes). To further challenge the classification model, most of the non-fire images include fire-like coloured objects. The process of image augmentation is used in order to multiply the number of images in the dataset.

Use Google Colab platform (colab.research.google.com) to build, train, and validate the CNN model. Google Colab provides free access to cloud GPUs for faster model training.

5.3. Recommendations

- Based on the obtained promising results, this study recommends implementing the proposed computer-vision based fire detection system for indoor and outdoor applications in the real world. Especially, the system does not require a high initial investment, since it will run on top of the existing infrastructure of the digital security cameras & surveillance system network.
- As a kick-off step, this thesis recommends using the proposed computer-vision fire detection system as a supplementary or added layer of protection to the existing conventional point-type fire detectors used in indoor applications.
- To improve the accuracy of the machine-learning based fire and smoke detection model, the thesis recommends acquiring better and more images that represents real fire in its early stages.
- The thesis recommends using YCbCr colour filter with the threshold values indicated in chapter-3 for efficient colour-segmentation of the fire region.
- The thesis recommends using the median background subtraction to extract foreground model of fire region.
- The thesis recommends the use of deep learning algorithms to automate the process of feature extraction and object detection in image processing applications.
- This study proves that using a small number of layers of the CNN model and a small filter size in each layer can improve the performance and generalizability of the image classification model.
- The thesis recommends using the dilated convolution to extract more global features without increasing the kernel size.

- The thesis recommends using Google Colab platform for the CNN model training, fast experimentation, and research purposes.
- The work presented various performance metrics that can be best used for the evaluation of the image classification model.

5.4. Future Research

The collected datasets are quite small, making it extremely challenging to create high accuracy models. The datasets are not representative of the problem in a correct manner. Many of the images in the dataset contained examples of professional photos captured by news reports. Real world fires may look different.

Future efforts in fire detection research should focus more on the actual dataset gathering and categorizing process to ensure the dataset better represents how fires start in the real world, smolder, and spread in natural scene images.

Further research can be conducted in the future to continue improving the early fire detection system based on the work presented in this thesis.

Future research may include the following topics:

- Enhancing the classification accuracies when incorporating bright fire-coloured moving objects like vehicle’s headlights.
- Fire detection using moving camera
- Monitoring the fire growth and the direction of spread.
- Camera sensitivity analysis.
- Improvements of the CNN image classification and object detection models.
- Better data collection methods.
- Real-life experimentation.
- Wind direction analysis.
- Fire spread analysis.
- Fuzzy logic extension.
- Agent-based simulation for evacuation.
- Link to fire departments and emergency management officials.
- Suggest fire extinguishing method depending on situation.
- Comparison with other commercial systems.

References

1. C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, “The Quickhull Algorithm for Convex Hulls,” ACM Transactions on Mathematical Software, vol. 22, no. 4, pp. 469–483, 1996
2. Candle Sparks a Serious House Fire || ViralHog, https://www.youtube.com/watch?v=7Awg8RAC-Xs&ab_channel=ViralHog (accessed in December 2021)
3. CBC New: Massive fire on property of Headingley business, Manitoba https://www.cbc.ca/news/canada/manitoba/headingley-best-buy-homes-fire-1.5854952 (accessed in December 2021)
4. T Celik, H Demirel, “Fire detection in video sequences using a generic colour model” - Fire safety journal, 2009 – Elsevier
5. CTV News: Massive fire breaks out in townhouse complex in Toronto, Ontario https://toronto.ctvnews.ca/massive-fire-breaks-out-in-townhouse-complex-in-toronto-smoke-drifts-over-highway-401-1.5218573 (accessed in December 2021)
6. Y Cui, H Dong, E Zhou, “An early fire detection method based on smoke texture analysis and discrimination” - 2008 Congress on Image and Signal Processing, 2008 – ieeexplore. ieee.org
7. D Durmus - “CIELAB colour space boundaries under theoretical spectra and 99 test colour samples” - Colour Research & Application, 2020 - Wiley Online Library
8. P Foggia, A Saggese, M Vento, “Real-Time Fire Detection for Video-Surveillance Applications Using a Combination of Experts Based on Colour, Shape, and Motion” - IEEE Transactions on Circuits and Systems for Video Technology, Vol. 25, No. 9, September 2015
9. F Gong, C Li, W Gong, X Li, X Yuan, Y Ma, T Song, “A real-time fire detection method from video with multi-feature fusion,” Computational Intelligent and Neuroscience, 2019 - hindawi.com
10. Rafael C. Gonzalez, Richard E. Wood, “Digital Image Processing,” ch. 1, second edition, Pearson Prentice Hall, 2002
11. Google ML Crash Course | Classification: True vs. False and Positive vs. Negative, https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative (accessed in December 2021)

12. Z Huang, J Leng, “Analysis of Hu's moment invariants on image scaling and rotation” - 2010 2nd International Conference on Computer Engineering and Technology, 2010 - ieeexplore.ieee.org
13. Fire Dataset: Outdoor-fire images and non-fire images for computer vision tasks. https://www.kaggle.com/phylake1337/fire-dataset/ (accessed in December 2021)
14. DP Kingma, J Ba, “Adam: A method for stochastic optimization” arXiv preprint arXiv:1412.6980, 2014 - arxiv.org, www. arxiv.org/abs/1412.6980
15. A Krizhevsky, I Sutskever, G.E. Hinton, "ImageNet classification with deep convolutional neural networks". In Advances in Neural Information Processing Systems 25; Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.
16. CB Liu, N Ahuja, “Vision based fire detection,” - Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004 - ieeexplore.ieee.org
17. MathWorks® Help Center | Fuzzy c-means clustering https://www.mathworks.com/help/fuzzy/fcm.html (accessed in December 2021)
18. S Miyamoto, H Ichihashi, K Honda, H Ichihashi, “Algorithms for Fuzzy Clustering: Methods in c-Means Clustering with Applications” - 2008 - Springer
19. NFPA's "Fire Loss in the United States During 2020" – NFPA report Sep. 2021, NFPA report - Fire loss in the United States (accessed in December 2021)
20. A Oliva, A Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope" - International journal of computer vision, 2001 – Springer. 8 Scene Categories Dataset https://people.csail.mit.edu/torralba/code/spatialenvelope/ (accessed in December 2021)
21. Ontario Ministry of the Solicitor General | Fire Losses: Causes, Trends, Issues from 2010 to 2019, Fire Losses: Causes, Trends, Issues | Ministry of the Solicitor General (gov.on.ca) (accessed in December 2021)
22. David M W Powers “Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation” - arXiv preprint arXiv:2010.16061, 2020 - arxiv.org
23. C Poynton - “Digital Video and HD: Algorithms and Interfaces” - 2012 - books.google.com - YUV and luminance considered harmful

24. C. E. Premal, S. S. Vinsley, “Image processing based forest fire detection using YCbCr colour model” - 2014 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2014]
25. Paul Read; Meyer, Mark-Paul; Gamma Group (2000). Restoration of motion picture film. Conservation and Museology. Butterworth-Heinemann. pp. 24–26. ISBN 978-0-7506-2793-1.
26. Ready.gov | Home Fires: Learn About Fires, www.ready.gov/home-fires - An official website of the United States government (accessed in December 2021)
27. Bernhard Scholkopf, Alexander J. Smola, “Learning with Kernels Support Vector Machines, Regularization, Optimization, and Beyond,” 2018
28. J. Seebamrungsat, S. Praising, and P. Riyamongkol – “Fire detection in the buildings using image processing” - 2014 Third ICT International Student Project Conference (ICT-ISPC2014) - ieeexplore.ieee.org
29. A Sharma, P Kumar Singh, Y Kumar, “An integrated fire detection system using IoT and image processing technique for smart cities” - Sustainable Cities and Society, 2020 - Elsevier
30. Stanford Vision and Learning Lab | Course CS231n: Convolutional Neural Networks for Visual Recognition, Spring 2021, https://cs231n.github.io/convolutional-networks/ (accessed in December 2021)
31. S Surit, W Chatwiriya, “Forest fire smoke detection in video based on digital image processing approach with static and dynamic characteristic analysis” - 2011 First ACIS/JNU International Conference on Computers, Networks, Systems, and Industrial Engineering, 2011 - ieeexplore.ieee.org
32. TechCrunch: Google’s AlphaGo AI beats the world’s best human Go player | TechCrunch, https://techcrunch.com/2017/05/23/googles-alphago-ai-beats-the-worlds-best-human-go-player/ (accessed in December 2021)
33. The Denver Channel: Massive fire destroys 2 buildings under construction in Denver https://www.thedenverchannel.com/news/local-news/massive-fire-destroys-building-under-construction (accessed in December 2021)
34. BU Toreyin, Y Dedeoglu, AE Cetin, “Contour based smoke detection in video using wavelets”- 14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8,2006, 2006 - ieeexplore.ieee.org

35. BU Töreyin, Y Dedeoğlu, U Güdükbay, "Computer vision based method for real-time fire and flame detection" - Pattern recognition letters, Volume 27, Issue 1, 1 January 2006 – Elsevier
36. T.X. Truong, J.M. Kim, "Fire flame detection in video sequences using multi-stage pattern recognition techniques" - Engineering Applications of Artificial Intelligence, 2012 - Elsevier
37. Vipin V, "Image Processing Based Forest Fire Detection" - International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 2, February 2012)
38. D Wang, X Cui, E Park, C Jin, H Kim, "Adaptive flame detection using randomness testing and robust features," - Fire safety journal, 2013 – Elsevier
39. Wikipedia | Wavelet packet decomposition https://en.wikipedia.org/wiki/Wavelet_packet_decomposition (accessed in December 2021)
40. A.K.K. Wong, N.K. Fong - Experimental study of video fire detection and its applications - Procedia engineering, 2014 – Elsevier
41. Fisher Yu, Vladlen Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions" Submitted on 23 Nov 2015 (v1), last revised 30 Apr 2016 (v3)
42. F Yuan, 'Video-based smoke detection with histogram sequence of LBP and LBPV pyramids"- Fire Safety Journal - Volume 46, Issue 3, April 2011, Pages 132-139, 2011 - Elsevier
43. F Yuan, "A double mapping framework for extraction of shape-invariant features based on multi-scale partitions with AdaBoost for video smoke detection" - Pattern Recognition, Volume 45, Issue 12, December 2012, Pages 4326-4336, 2012 - Elsevier
44. D Zhang, Y Rao, J Zhao, J Zhao, "Feature based segmentation and clustering on forest fire video," - 2007 IEEE International Conference on Robotics and Biomimetics (ROBIO) - ieeexplore.ieee.org