Yarmouk Private University

Faculty of Informatics and Communication Engineering

Department of Software Engineering

Utilizing of Deep Learning Model for Real Time Fire Detection

Applied Project

By:

MHD ADEL MOMO

Supervisor:

Dr.Nouar AlDahoul

Semester 2018/2

Abstract

Fire problem has been the target for researches since long time ago. Different devices, sensors and methods have been proposed and implemented to mitigate the effects of fire by detecting it in early stages. Early detection helps to stop fire spread to save lives and mitigate the destruction of the infrastructure. Traditional sensors such as smoke, heat, infrared, ultraviolet light radiation, and gas have been used for fire detection. These sensors are sometimes not efficient and suffer from some problems such as low sensitivity and high false alarm. This project proposes a deep learning model that utilizes Convolutional Neural Network (CNN) for visual fire detection task. A low-cost surveillance camera is used as a sensor to capture a stream of visual data (images) from the surrounding. These images are applied to CNN model for feature learning and then mapped to two classes: fire and non-fire classes. The model is implemented on low cost processing platform such as raspberry pi. The results are promising with average accuracy of 92.6 %. The proposed fire detection system can detect fire in real time and send warning message to people in charge to avoid the troubles. The objective of the proposed system is to increase the detection sensitivity by correctly classifying the fire scene as a fire and at the same time reduce the high false alarm that results from false detection of normal scene as a fire.

Table of Contents

List of Figures

List of Tables

Chapter (1)

Introduction to Project

1.1. Introduction

Fire spread phenomenon is increasing especially in the forests, homes, buildings due to various reasons. Some of these reasons are high temperature degree, short circuit, …. etc. Due to the fire spread phenomenon, fire sensors are raised to detect the fire in home building, airports, banks, hospitals, facility, and other places. In this project, we propose a fire detection system that consists of two components:

- Fire classification: This component detects a fire by a surveillance camera and send warnings to a user once a fire is detected.
- Android application such as WhatsApp: This component receives the warning from the fire classification stage.

1.2. Problem Statement

Fire problem has been the target for researches since long time ago. Early detection helps to stop fire spread to save lives and mitigate the destruction of the infrastructure. Traditional sensors such as smoke, heat, and gas are sometimes not efficient and suffer from few problems such as low sensitivity and high false alarm. Low sensitivity problem raises from misclassifying fire scene as a normal one. High false alarm results from misclassifying a normal scene as a fire one. Feature engineering methods have been used to extract features from data before classification stages. These methods are domain dependence and require experts to select good algorithms for hand-crafted features extraction.

1.3. Project Objective

The objectives of this project are:

- To design a deep neural network model and to fine tune the hyper parameters in order to achieve good results in term of the accuracy and speed.
- To implement the previous detection system on low cost processing platform such as raspberry pi to detect a stream of visual data captured by that camera which is attached to raspberry pi.
- Design and implement an API to send a warning message to WhatsApp and SMS.

1.4. Project Block Diagram

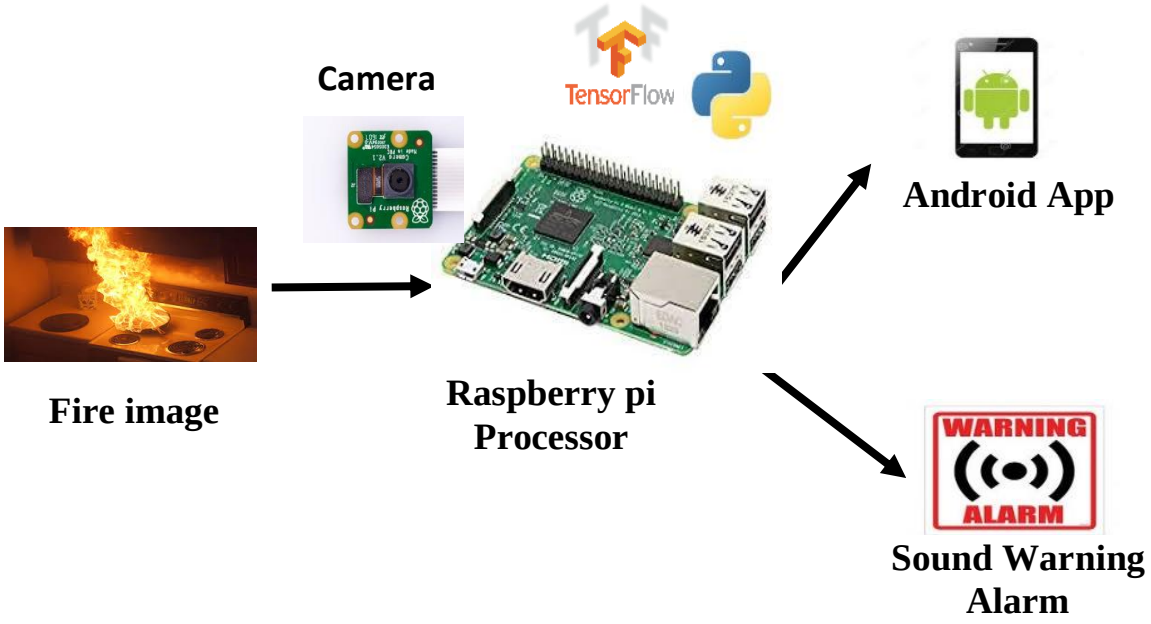

Figure 1.1. Block Diagram

1.5. Methodology

The algorithms that are used in our project are:

- Convolution Neural Network: Convolutional neural network has 3 major components:
    - Convolution layer: which takes an image / multi-dimensional matrix as an input and convolves this matrix with a kernel, the output is a multi-dimensional matrix.
    - Max pool layer: This layer is responsible to reduce the number of features / reduce the dimensions of the multi-dimensional matrix by applying a kernel with a certain size to the output of the convolution layer which is a multi-dimensional matrix, the output is the reduced multi-dimensional matrix.
    - Fully connected layer: This layer is the final layer, it takes the multi-dimensional matrix as an input, the output is the prediction whether there is a fire or not.

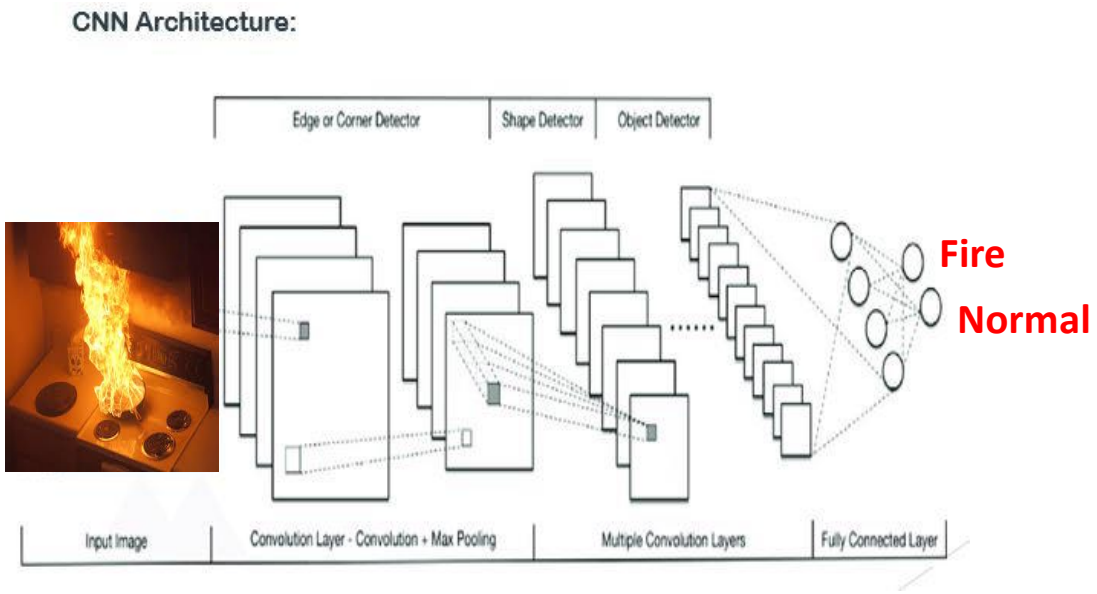

Figure 1.2. Convolutional Neural Network Architecture

Chapter (2)

Literature Review

2.1. Literature Review

We studied some research papers that are related to our project which is fire detection and we have found their solutions below.

- DESIGN AND ANALYSIS OF AN INTELLIGENT FIRE DETECTION SYSTEM FOR AIRCRAFT [2]: This research paper proposes a solution to fire detection in aircraft. Recently, many fire detectors are proposed to detect a fire in aircraft which are thermal sensors, smoke sensors. This paper has found an alternative solution to the sensor devices. The paper's solution is based on "Artificial Neural Network", the "ANN" is a 3 layers network so it is a shallow "Neural Network", the "ANN" takes 3 input variables which are:
    - The temperature.
    - The smoke density.
    - The $O2$ gas.

    The network' structure.

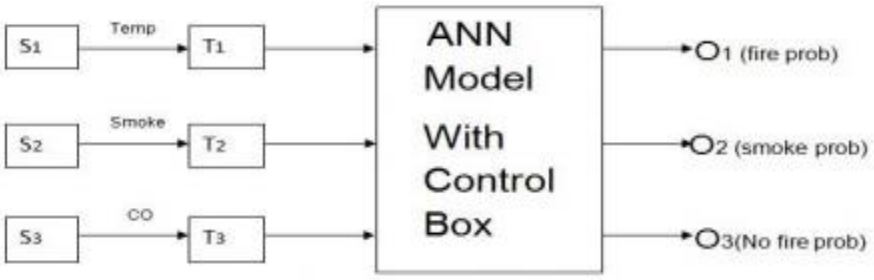

Figure 2.1. Neural Network Architecture

The network takes 3 input numeric variables as input and tries to predict 3 output numeric values.

- AUTOMATIC FIRE DETECTION: A SURVEY FROM WIRELESS SENSOR NETWORK PERSPECTIVE [1]: This paper discusses several solutions to the fire detection problem, the solutions are:
    - Fire detection based on WIFI sensors, this solution uses sensor nodes and task manager, each node uses temperature, humidity sensors, the decision making is done by the task manager.
    - Fire detection based on "Artificial Neural Network", this solution is similar to the previous paper.
- An Efficient Deep Learning Algorithm for Fire and Smoke Detection with Limited Data [3]: This solution proposes a "Convolutional Neural Network" architecture that is appropriated with a limited dataset in term of the number of samples in the dataset. This paper is focused on classifying whether an image is fire or smoke or background which means no fire neither smoke. The authors of the paper have suggested a new architecture of the "Convolutional Neural Network" with a new activation function in the hidden layers of the network "Adaptive Piecewise Linear", the authors found that using this activation function in the hidden units of the network improved the performance of the network in term of accuracy. The architecture of the network is:

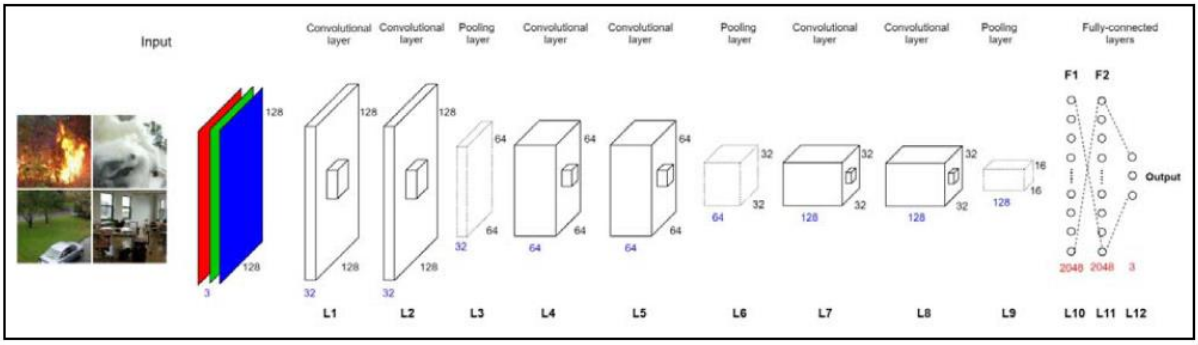

Figure 2.2. Paper's Network Architecture

The network of the paper has overfitting which means the network has high accuracy in the training data but in the test data has poor performance, so the authors used the "Generative Adversarial Network", this type of network generates new sample from the data set distribution through giving the network random matrix as an input and the network generates from this random a fire image for example. The "Generative" network has 2 networks which are:

- Generator: This neural network is taking a random image as an input and generates from this random a new sample from the same training set distribution.
- Discriminator: This neural network is taking the generated sample from the generator and classify whether the sample is generated or real, the discriminator trains the generator to generate the samples to be as same as the training data.

The figure in the right shows a generated sample in another weather condition from the left sample:

Figure 2.3. Generated Sample

The authors have generated a new dataset from the "Generative" model which lead to a high performance.

- FireNet: A Specialized Lightweight Fire & Smoke Detection Model for Real-Time IoT Applications [4]: This paper proposes an alternative solution to the solutions based on sensor devices which depends on the flames. The solution of the authors is deploying a shallow "Convolutional Neural Network" that runs on a IOT node in real time connected to a surveillance camera. The IOT device that is used is "Raspberry Pi" which is connected to cloud server in order to send alarm messages to the mobile application via the cloud. The architecture of the IOT system is:

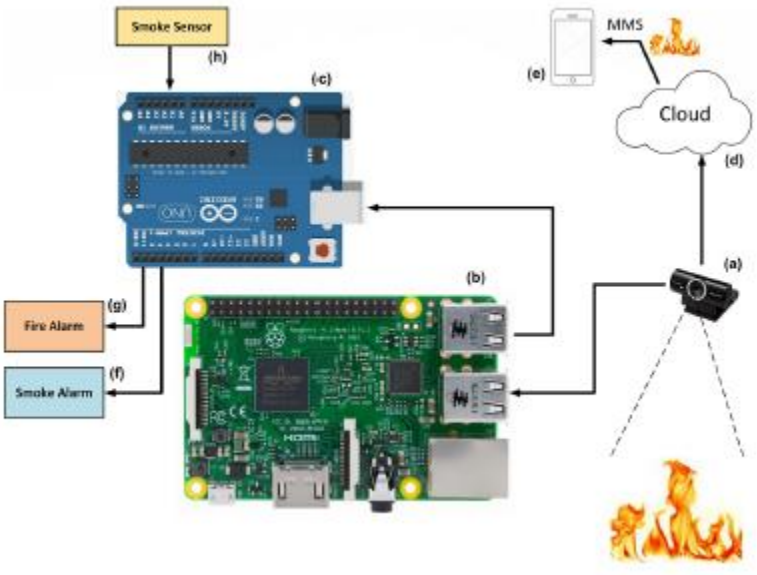

Figure 2.4. Paper's System

The accuracy of their model is:

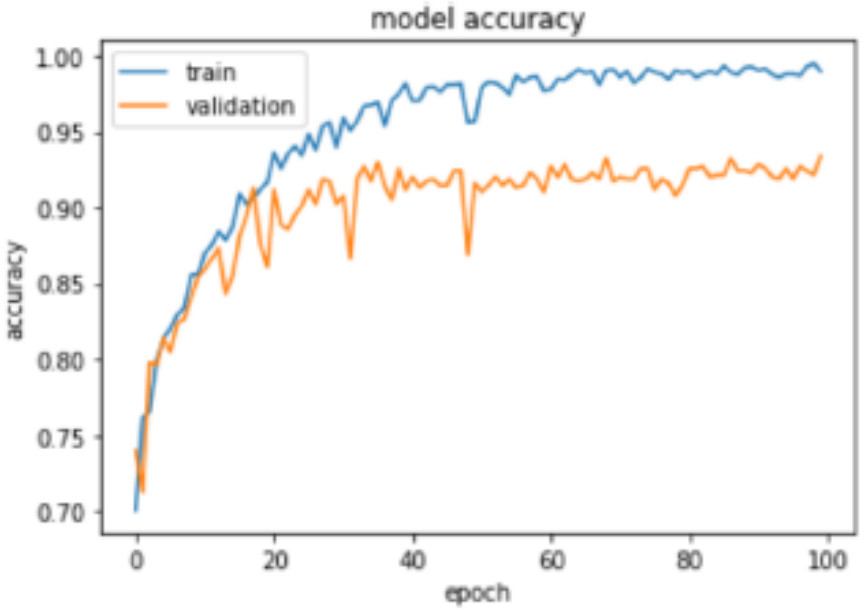

Figure 2.5. Network Accuracy

The architecture of the shallow "Convolutional Neural Network" of the paper is:

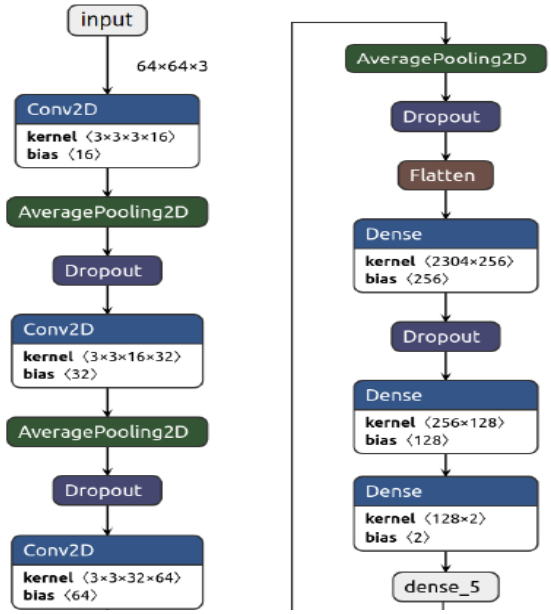

Figure 2.6. Neural Network Architecture

The reason of using shallow network in term of the number of the parameters is to be able to run the network in real time on the embedded device up to 30 frames per second. The performance of the sparse model is summarized in the table:

Metrices	Our dataset (%)	Foggia's dataset (%)
Accuracy	**93.91**	**96.53**
False Positives	1.95	1.23
False Negatives	4.13	2.25
Recall	94	97.46
Precision	97	95.54
F-measure	95	96.49

Figure 2.7. Network Performance Metrics

- Convolutional Neural Networks Based Fire Detection in Surveillance Videos [5]: This paper proposes a pretrained "Convolutional Neural Network" model similar to the "GoogleNet" deep model in fire and flame detection, the paper proposes to apply the model on a powerful embedded device that is connected to a surveillance video camera to detect fire or normal situation in real time. The paper proposes their network that the deep network model will be applied on a powerful device in term of computational complexity. The architecture of the system is:

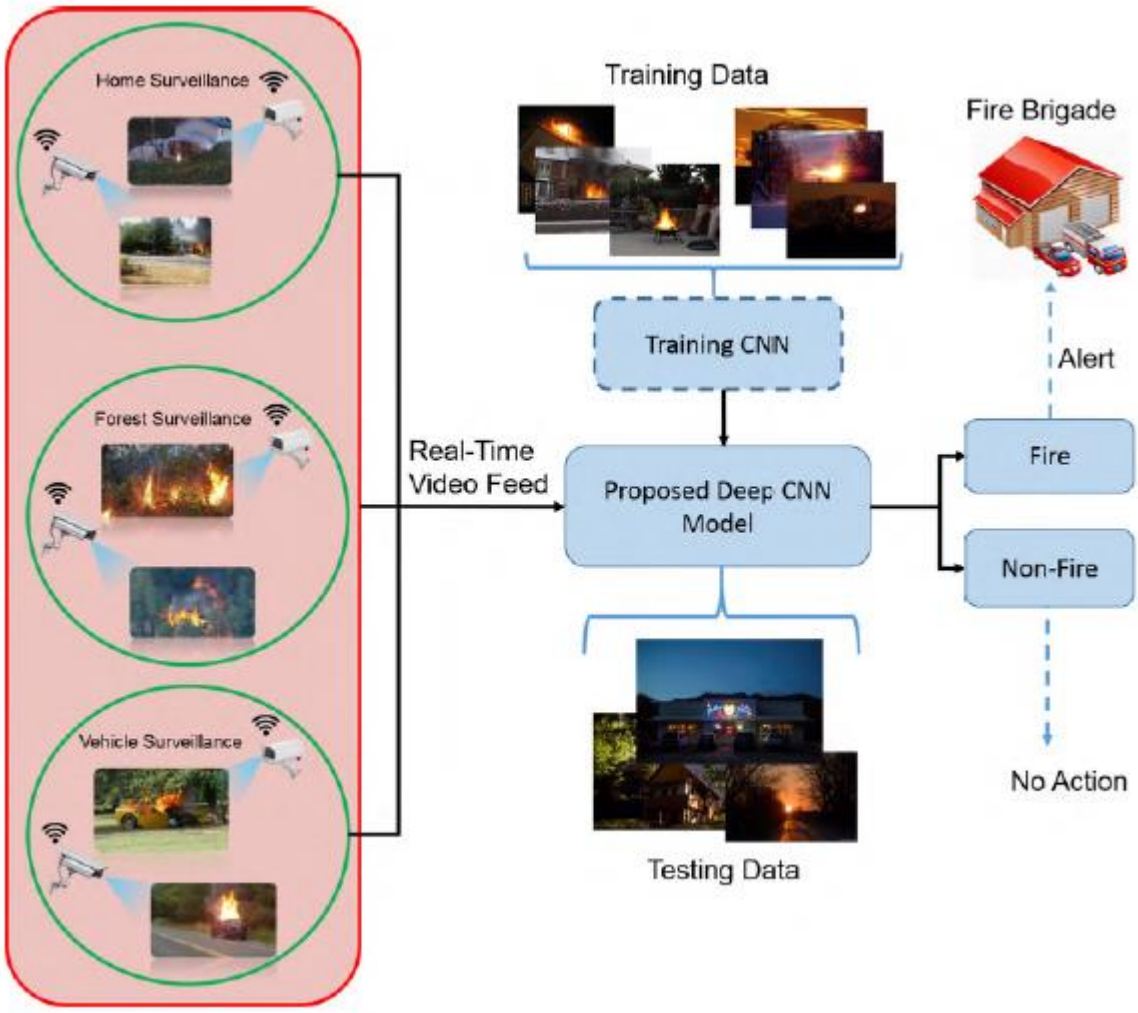

Figure 2.8. System Architecture

The architecture of the deep learning model of the authors is:

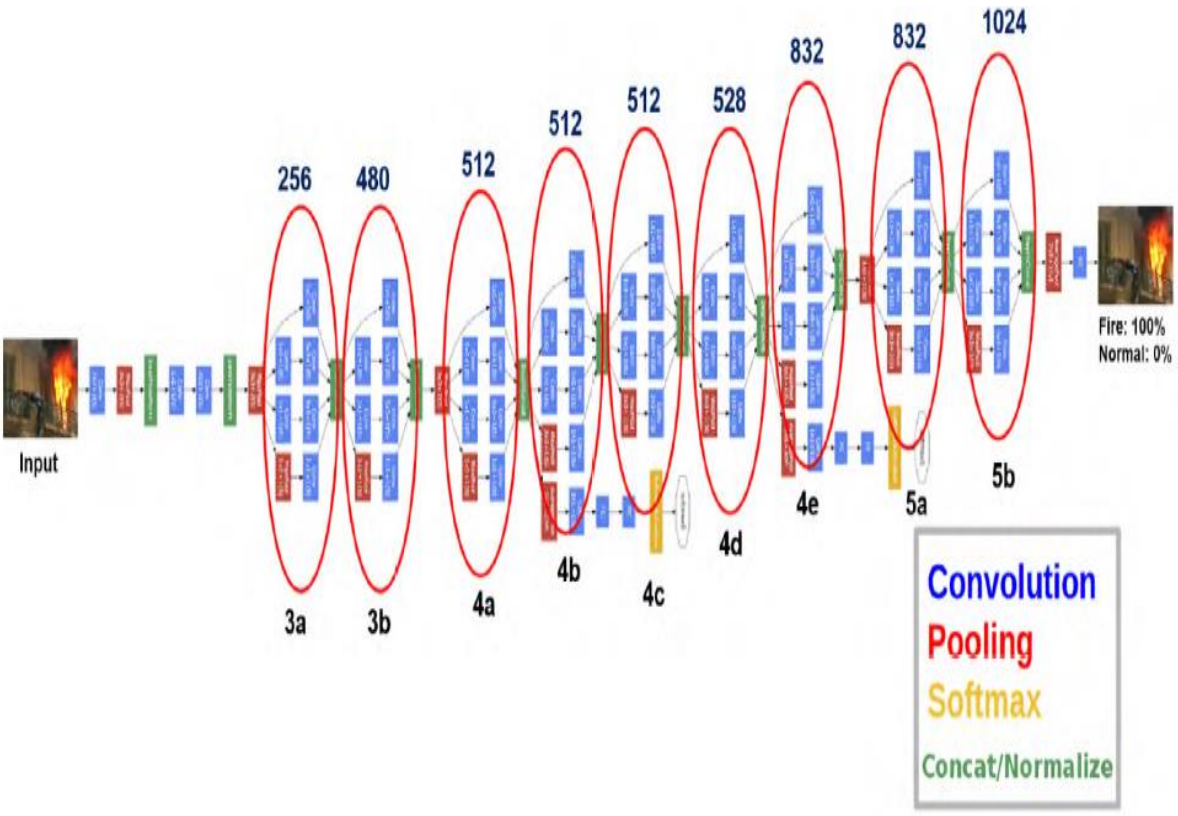

Figure 2.9. Paper's Network Architecture

The model's results on test images are:

Figure 2.10. Model's Results

Some of paper's dataset samples:

Figure 2.11. Paper's Dataset Samples

Chapter (3)

Machine Learning

for Fire Detection

3.1. Fire Detection Application in a Facility

Figure 3.1. Raspberry Pi and Camera

The above component is installed in somewhere in a facility to monitor the area. It includes "Raspberry Pi" processor and the camera as shown in figure 3.1, 3.2.

Figure 3.2. Raspberry Pi and Camera in a Facility

3.2. Feature Engineering

Feature engineering is to apply a set of mathematical equations to a dataset and then to extract the quantity of each mathematical formula for each sample in the dataset. After creating a new dataset from the old one, the new data is passed to a simple machine learning algorithm such as “Logistic Regression”. For fire dataset application, a lot of computer vision algorithms have been applied to extract attributes from the images.

3.3. Feature Learning

The new trend now is the deep neural networks. Deep neural networks have a variant of architectures. The fire detection is a computer vision task. The Convolutional Neural Network is the state of the art in the computer vision tasks. Convolutional Neural Network learns the features of the dataset automatically through the convolution layer in figure 3.3.

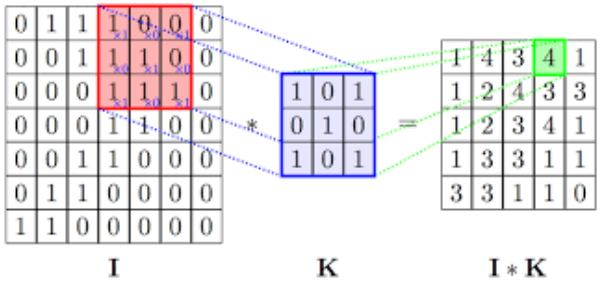

Figure 3.3. Convolution Layer

The Convolutional Neural Network is powerful tool for computer vision applications. It solves these types of problems easier. On the other hand, traditional solutions that depend on extracting features give sometimes poor accuracy and at the same time they require experts in the domain of the application.

3.4. Computational Graph

Computational Graph is a set of nodes and a set of edges. Each edge can be considered as a multi-dimensional matrix / tensor what we called, each node is a mathematical operation such as convolutional layer or max pooling layer or activation function. A neural network is a computational graph that is each layer is a node and each edge is a tensor / multi-dimensional matrix. When we build a deep model, we build our computational graph. After building the graph, we should think in how can we optimize / train our computational graph, we should use the optimization algorithms to optimize / train our graph through minimizing the error of our graph such "Gradient Descent" see the computational graph of our project in figure 3.4. To train our graph, we must specify a learning rate "0.01" traditionally and pass it to the optimization algorithm, then when the graph compute the final node which means the output layer, then the "Backpropagation" algorithm computes the gradients of the nodes and edges in the graph with respect to the loss / error then the graph updates its edges through the "Gradient Descent" optimization procedure in order to minimize the error through this basic formula:

$$w := w - \alpha \frac{\partial error}{\partial w}$$

A computational graph reduces the loss through the formula that is given above in a iterative approach which means the graph is fed with a batch of samples in the training set such as 64 batch size then apply the "Backpropagation" to that batch and updates its parameters using the "Gradient Descent" with respect to that error's batch and keep feeding batches until the last batch and start from the first batch and keep training and so on.

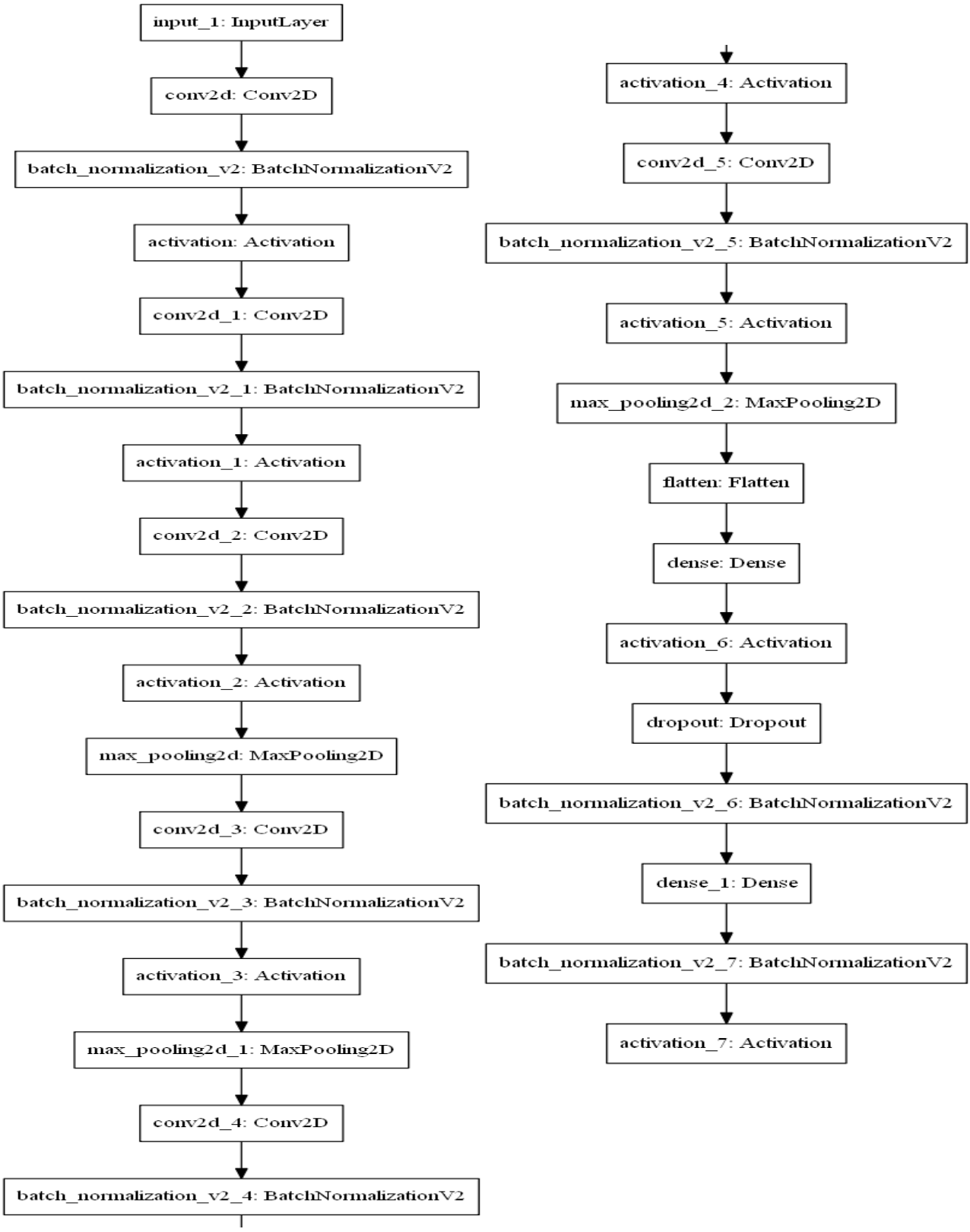

Figure 3.4. Our Computational Graph

Chapter (4)

Experimental Results and Analysis

4.1. Dataset Preparation

First, we have searched for fire datasets in different web sites and academic papers, but we didn't find ready one that fits our requirement. we decided to collect our own fire dataset. First, we started emailing a research group which has published an academic research paper in fire detection topic and asked them to send us their dataset. After getting the dataset, we continued collecting our data from fire images and videos available in YouTube and google. The number of samples in total is 1891 sample with both fire and non-fire images. Table 4.1 describe the number of training and testing samples.

Table 4.1. Training and Testing Samples

Training Samples	1323
Testing Samples	568
Total	1891

The images are RGB colored. To train our neural network, the dataset needs to be normalized / standardized. The dataset is standardized by the formula:

$$X(i,j) = (X(i,j) - \mu) \quad / \quad \sigma$$

where:

$X(i,j)$: The pixel element in the $ith\ row\ and\ jth\ column.$

μ : The mean of the pixel in the $X(i,j)$ location across the whole dataset.

σ : The standard deviation of the $X(i,j)$ pixel intensity.

Standardization is an essential processing step in order to train the network faster where the loss / error of the network is converging to the minimum in a fast way.

The following figures have few samples of fire and non-fire images in the dataset:

Figure 4.1. Fire Sample 1

Figure 4.2. Fire Sample 2

Figure 4.3. Fire Sample 3

Figure 4.4. Non-Fire Sample 1

Figure 4.5. Non-Fire Sample 2

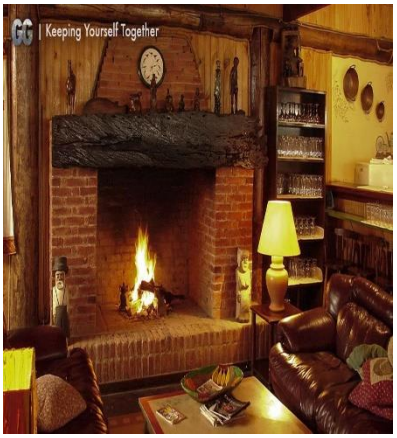

Figure 4.6. Non-Fire Sample 3

Figure 4.7. Non-Fire Sample 4

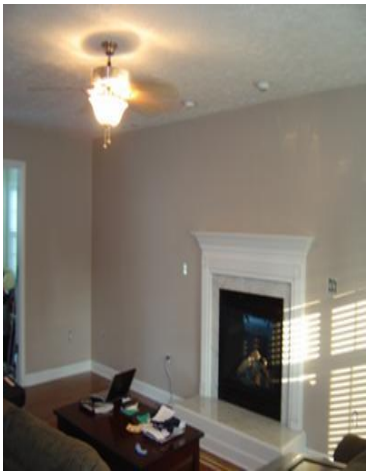

Figure 4.8. Non-Fire Sample 5

Figure 4.9. Non-Fire Sample 6

4.2. Experimental Results

The following figures are the few samples of images applied to the proposed deep learning model for fire / non-fire detection. The probabilities of detection are printed in blue color.

Figure 4.10. Experiment 1

Figure 4.11. Experiment 2

Figure 4.12. Experiment 3

Figure 4.13. Experiment 4

Figure 4.14. Experiment 5

Figure 4.15. Experiment 6

Figure 4.16. Experiment 7

Figure 4.17. Experiment 8

Figure 4.18. Experiment 9

Figure 4.19. Experiment 10

Figure 4.20. Experiment 11

Figure 4.21. Experiment 12

Figure 4.22. Experiment 13

Figure 4.23. Experiment 14

Figure 4.24. Experiment 15

Figure 4.25. Experiment 16

Figure 4.26. Experiment 17

Figure 4.27. Experiment 18

Figure 4.28. Experiment 19

Figure 4.29. Experiment 20

Figure 4.30. Experiment 21

Figure 4.31. Experiment 22

4.3. Model Evaluation Metrics

To evaluate and anal size our results, performance metrics are used as following:

- Model Learning Curve.
- Model Loss Curve.
- Confusion Matrix.
- K-Fold cross validation.

The performance metrics of our proposed deep learning model are shown in these figures.

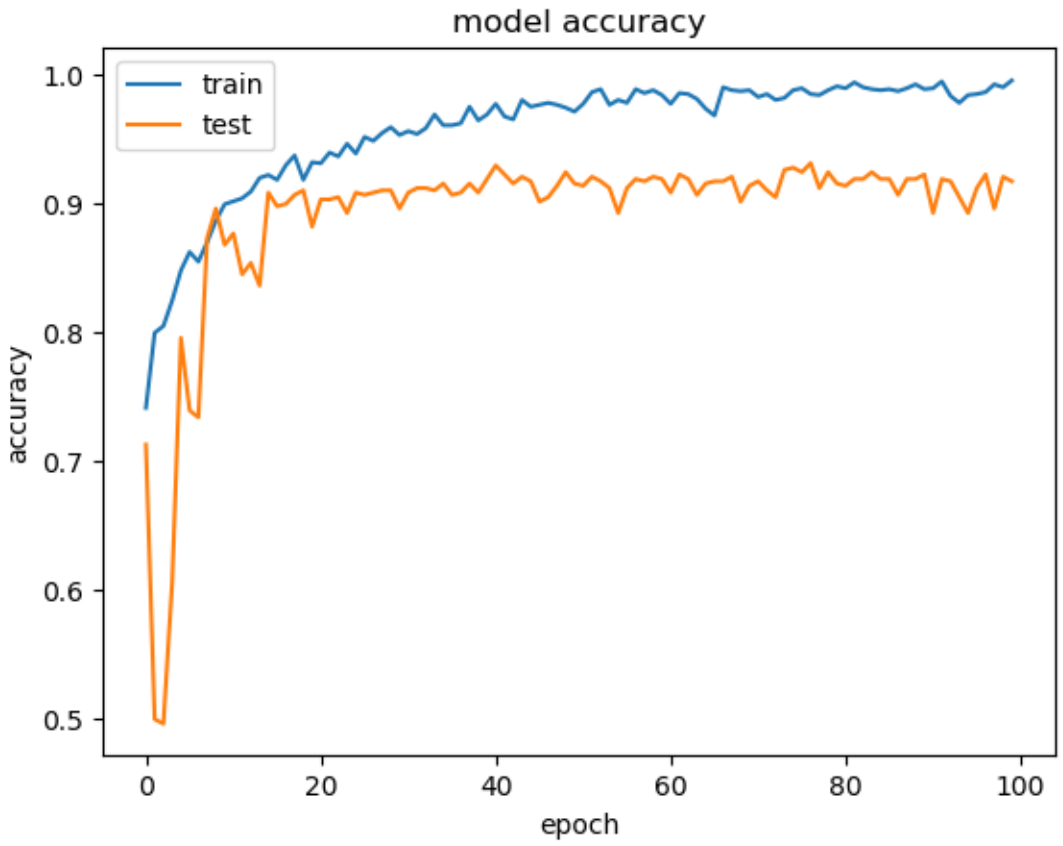

Figure 4.32. Learning Curve in Term of Accuracy

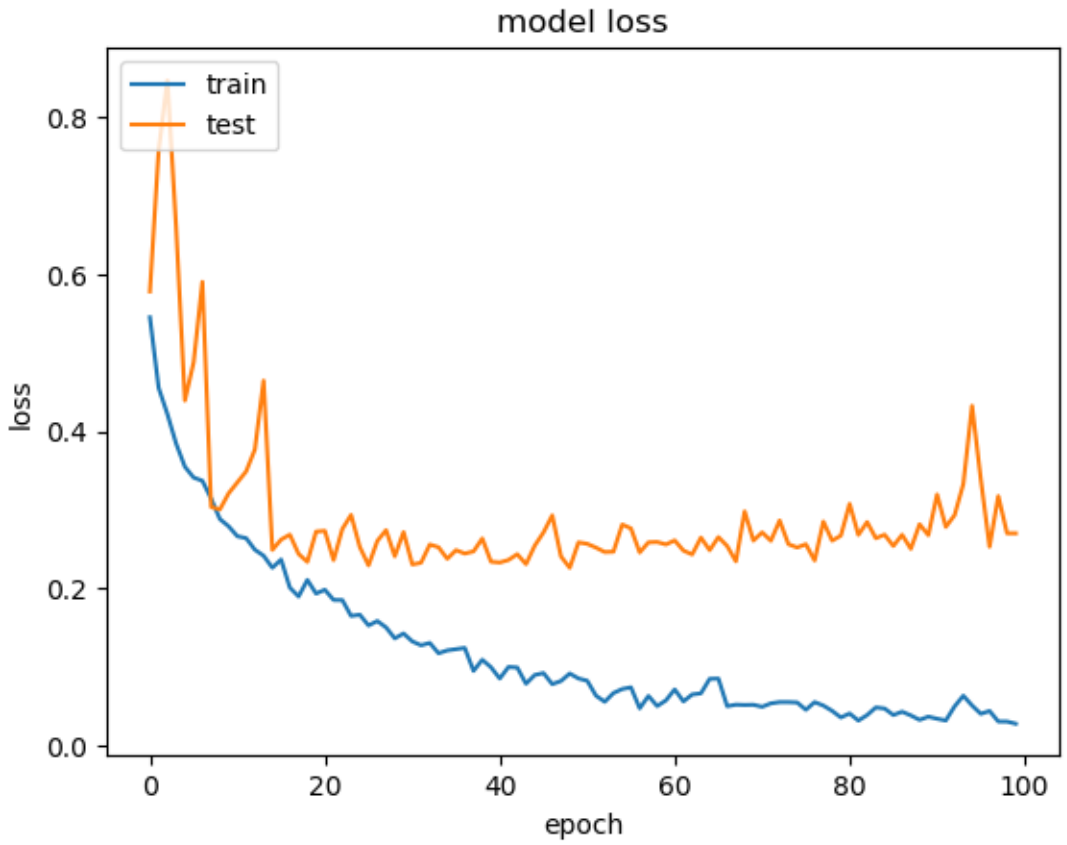

Figure 4.33. Learning Curve in Term of Loss

The objective function is the function that the neural network targets to minimize. Cross Entropy function, is used as a loss function with the following formula:

$$L = -Yi\ Log_2(Pi)$$

where:

Yi : is the label of the i_{th}sample in the training set.

Pi : is the predicted output of the i_{th} sample in the training set.

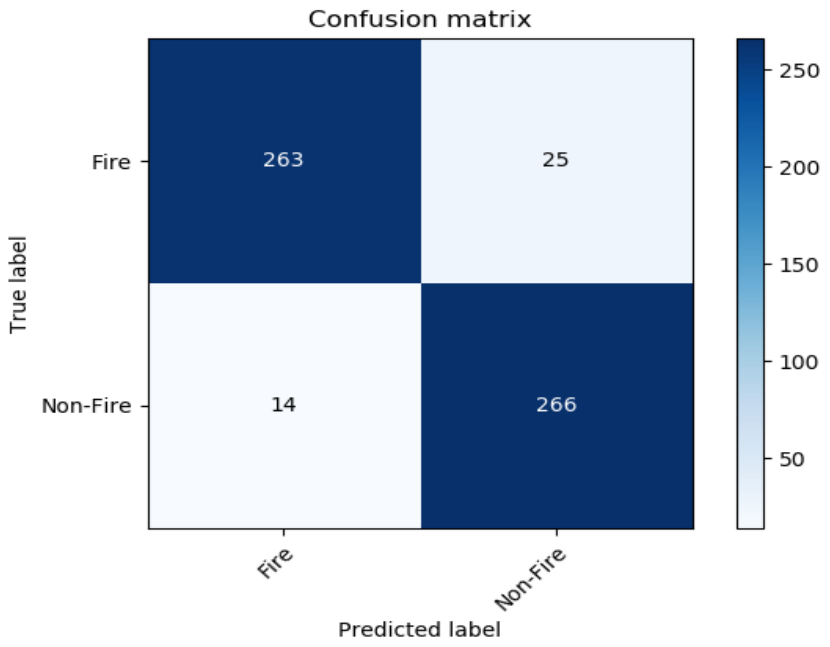

Figure 4.34. Confusion Matrix

Predictive Model: Evaluation

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

predictive result / classification		actual result / classification: yes	actual result / classification: no
	yes	**tp** (true positive)	**fp** (false positive) ← Type 1 error
	no	**fn** (false negative)	**tn** (true negative)

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \qquad \text{True Negative Rate} = \frac{tn}{tn + fp}$$

Figure 4.35. Another Model Evaluation Metrics

K-Fold cross validation: This type of testing is to split the dataset to “K” segments. K experiments are performed. In each experiment, a segment is used as testing data and the rests are used as training data. Each experiment has an individual accuracy. After performing “K” experiments, the average accuracy of the “K” experiments is calculated.

In this project, our dataset is spited to 5 segments. One segment is considered as testing data and the rests of four segments are training data. The model is trained 5 times. The learning curves for the 5 experiments are shown below.

Experiment / Fold (1):

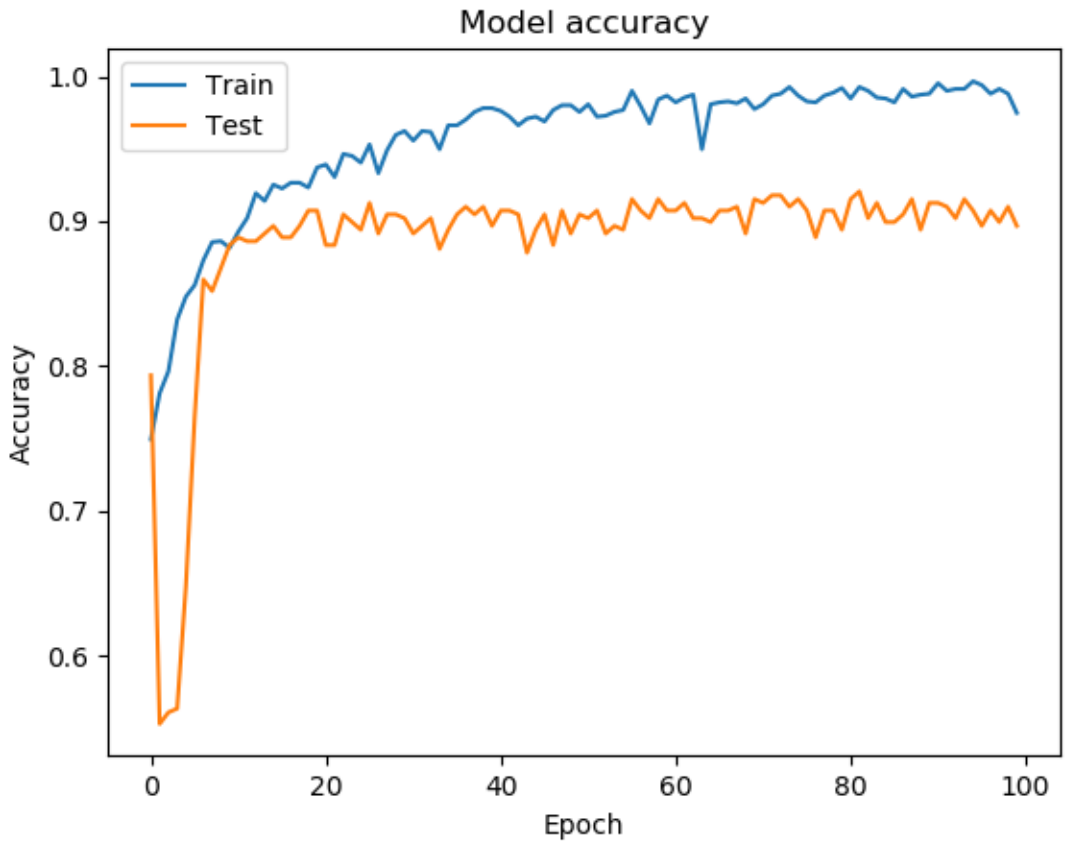

Figure 4.36. Learning Curve in Term of Accuracy of Fold (1)

Experiment / Fold (2)

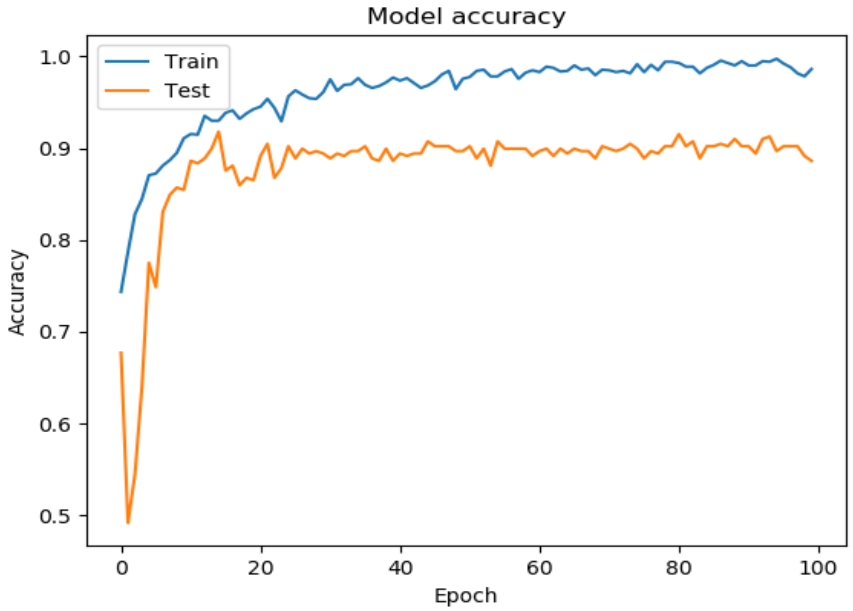

Figure 4.37. Learning Curve in Term of Accuracy of Fold (2)

Experiment / Fold (3)

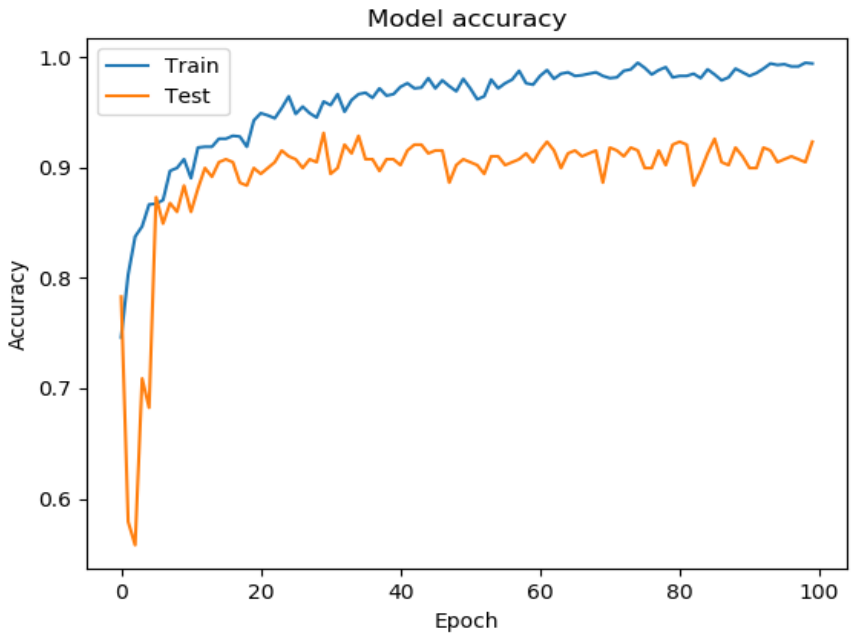

Figure 4.38. Learning Curve in Term of Accuracy of Fold (3)

Experiment / Fold (4):

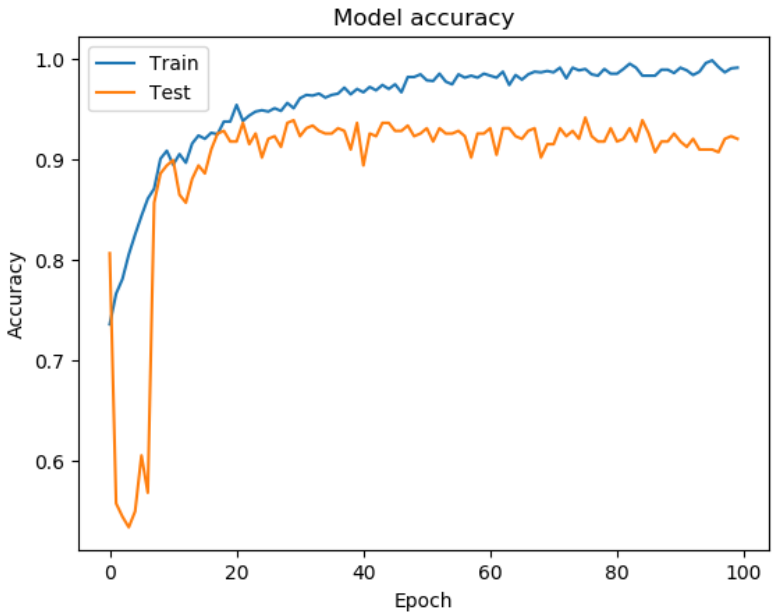

Figure 4.39. Learning Curve in Term of Accuracy of Fold (4)

Experiment / Fold (5):

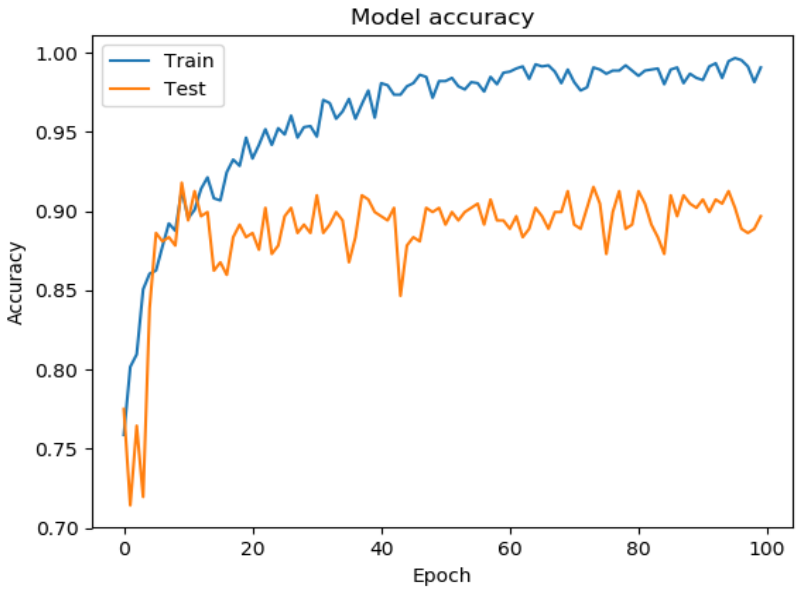

Figure 4.40. Learning Curve in Term of Accuracy of Fold (5)

Table 4.2. 5-Fold Cross Validation

Experiment	Accuracy
Experiment (1)	93.12%
Experiment (2)	91.79%
Experiment (3)	92.06%
Experiment (4)	94.18%
Experiment (5)	91.79%
Average	**92.59%**
Precision	**94.95%**
Call (True positive rate)	**91.32 %**
True negative rate	**95%**
F score	**93.099 %**

In each fold / experiment. we divided the dataset into two sets: the training and the testing set. The network is trained with training set. The network is evaluated with the testing set. The accuracy of each fold / experiment is recorded. The average accuracy of 5 folds is calculated as a final accuracy.

Chapter (5)

Internet of Things

(IOT)

5.1. Internet of Things (IOT)

IOT is a set of hardware nodes that can interact or talk with each other using the internet protocols such as “Transmission Control Protocol”. IOT is raised after the emergence of Cloud Computing. Cloud Computing supports us with storages, services, platforms, API functions…etc. A set of hardware devices in a facility or big city can interact with each other using the storage that the cloud companies provide. In our project we have used a “SMS” service that a cloud company provide to send cautions to a user when a fire is detected by the surveillance camera. The camera is connected to the “Raspberry Pi” device node in a facility. Without the Cloud Computing, it is hard to make the hardware nodes talk with each other. Cloud Computing is such a telephone between two devices. The hardware nodes talk through exchanging messages by using the internet protocols and the cloud storage. When the other device receives the message, it performs an action. The equation of the IOT is:

$$IOT = Cloud\ Computing + Hardware\ Nodes + Internet\ Protocols$$

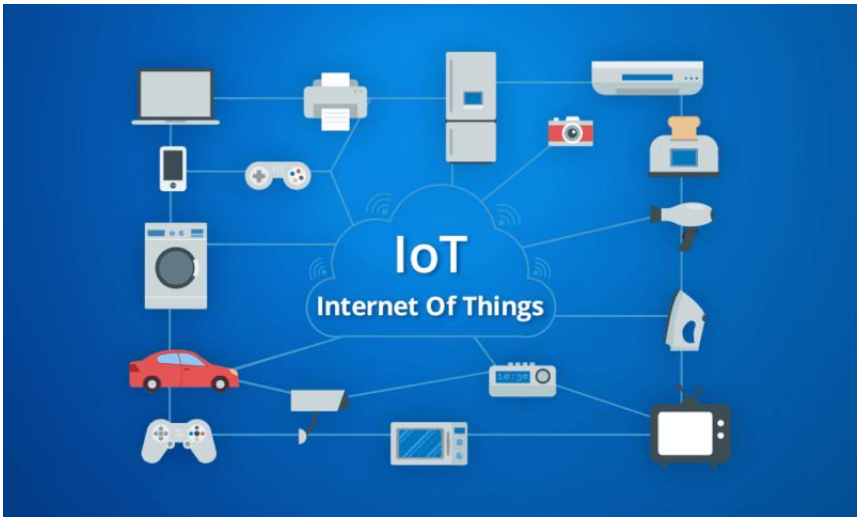

Figure 5.1. IOT Architecture

Chapter (6)

Software Diagrams and Frameworks

6.1. Use Case Diagram

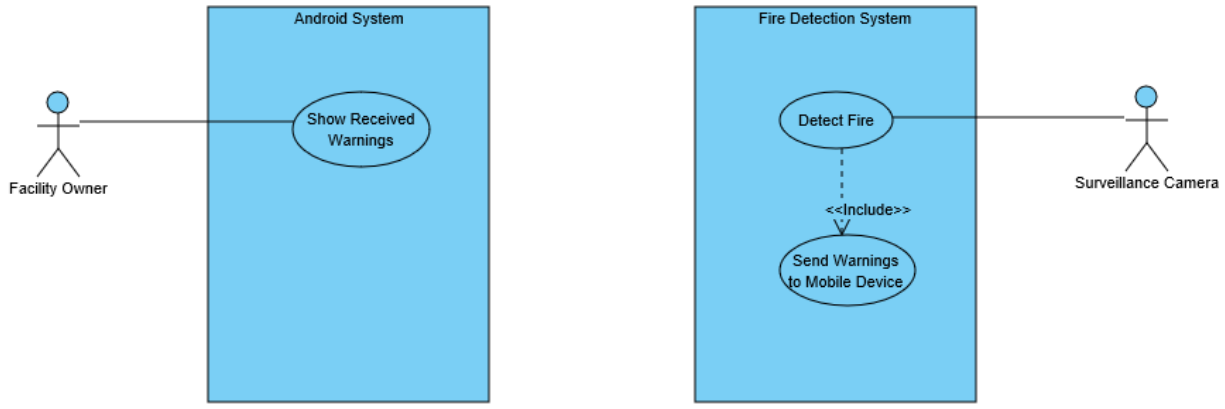

Figure 6.1. Use Case Diagram

6.2. Sequence Diagram

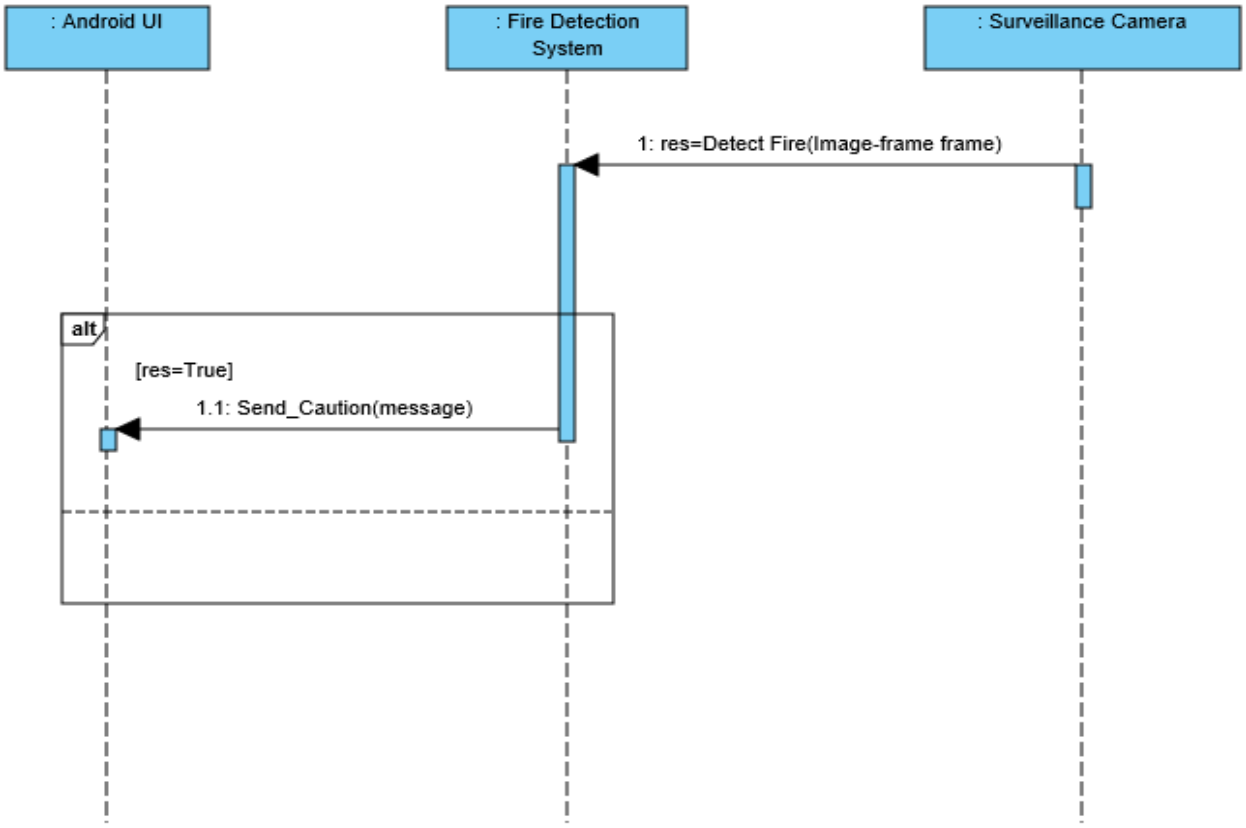

Figure 6.2. Sequence Diagram

6.3. Software Libraries

The libraries / frameworks that are used in our project are:

- Tensorflow: Tensorflow is a deep learning framework that focuses on building deep neural networks models in an efficient way. Tensorflow is used in our system in order to build a Deep Convolutional Neural Network model. This CNN model is used to detect whether a video frame has a fire or not.

Figure 6.3. Tensorflow Logo

- Opencv: is a computer vision framework that has the image processing and computer vision algorithms. Opencv is used in our system in order to open a video stream and read frames from that stream, a video stream is a loop process and has frame per second (FPS) measure, the standard frame rate of a video stream is 30 FPS.

Figure 6.4. OpenCv Logo

- Android-SDK: This framework is used for making an android application

Conclusion

We designed our system utilizing the deep learning model which is "Convolutional Neural Network", we have tested our model using the machine learning model metrics which are the k-fold cross validation, confusion matrix, we consider the following points to improve in the future:

- Our deep model has reached to [91-93] % accuracy, we are considering to improve the accuracy of the model.
- Design the system on another platform such as IOS operating system.

References

[1] Majid Bahrepour, Nirvana Meratnia, Paul Havinga,"AUTOMATIC FIRE DETECTION: A SURVEY FROM WIRELESS SENSOR NETWORK PERSPECTIVE", Pervasive Systems Group, University of Twente.

[2] Amit Yadav, Abhijeet Agawal, Pramod kumar, Tejaswi Sachwani,"DESIGN AND ANALYSIS OF AN INTELLIGENT FIRE DETECTION SYSTEM FOR AIRCRAFT", Communication, Integrated Networks & Signal Processing-CINSP 2018.

[3] Abdulaziz NAMOZOV, Young Im CHO, "An Efficient Deep Learning Algorithm for Fire and Smoke Detection with Limited Data", Volume 18, Number 4, 2018

[4] Arpit Jadon, Mohd. Omama and Akshay Varshney, "Detection Model for Real-Time IoT Applications", IEEE Region 10 Conference (TENCON), 2019

[5] KHAN MUHAMMAD, IRFAN MEHMOOD AND SUNG WOOK BAIK, "Convolutional Neural Networks Based Fire Detection in Surveillance Videos", IEEE Access, VOLUME 6, 2018