Fire Detection System Using Deep Learning Design Project Report at IIITDM Kancheepuram

Design Project report submitted in partial fulfilment of the requirements for the degree of B.Tech. (and M.Tech (for DD))

by

Student Arindam Sharma
(Roll No: CED17I022)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,
DESIGN AND MANUFACTURING, KANCHEEPURAM

December 2021

Certificate

I, **Arindam Sharma**, with Roll No: **CED17I022** hereby declare that the material presented in the Design Project Report titled **Fire Detection System Using Deep Learning Design Project Report at IIITDM Kancheepuram** represents original work carried out by me in the **Department of Computer Science and Engineering** at the **Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram** during the year **2021**. With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: 11-10-2021 Student's Signature Arindam Sharma

In my capacity as supervisor of the above-mentioned work, I certify that the work presented in this Report is carried out under my supervision, and is worthy of consideration for the requirements of design project work during the month November 2021.

Guide : **Dr. Pandiri Venkatesh**
Assistant Professor
Department of Computer Science and Engineering
IIITDM Kancheepuram

Abstract

This report contain the work done for design project which is undertaken in the fulfillment of a Course which is a part of my last Second Semester.working on this project lead me to learn lot of things .the purpose of this report is to build an application and understand the work flow behind an application .

This report is made of four chapters

- Chapter 1 : Introduction
- Chapter 2 : Prerequisites/Models
- Chapter 3 : Data-set
- Chapter 4 : Conclusion And Learning's

Acknowledgements

I would like to express my special thanks to **Dr. Venkatesh Pandiri** sir who help me a lot to understand and cleared most of the doubts ,even thought he have lot of other work and responsibilities ,he took out the time to help me out.

Contents

Abbreviations

CNN	**C**onvolution **N**eural**N**etwork
DL	**D**eep **L**earning
CV	**C**omputer **V**ision

List of Figures

Chapter 1

Introduction

There are lots of cases when surveillance shows fire hazards but nothing can you done as they are not equipped with the Fire Detection System ,there are only being alerted when fire alarm is active , there are currently lot of system which are equipped with visual bases system ,which can be used in general to stop fire hazards . Here i will discuss about the CNN mode to Detect fire using any visual equipped system (e.g surveillance system) here i have used the

1.1 Problem Statement

Able to Identify Fire Hazards at its initial stage and give it resolved with necessary help ,this setup can be installed in any visual based system. this will prevent major fire accidents.

The first model is a customized basic CNN architecture inspired by AlexNet architecture.

1.2 Motivation

Current system equipped with visual are most of the time used for Surveillance ,these system can be even used to detect fire based hazards even before it start ,like initial stage of fire .Current System for fire detection include smoke detectors and fire alarm ,but that have high latency as most of them are installed in the ceiling of the room .more the height of the room ,higher the latency of the devices will be.

To over come this issue ,here we can use the CNN based fire detector which are trained on more than thousand fire based images .

Chapter 2

DataSet

Preparing our Fire and Non-fire dataset involves a few process:

2.1 Directory Structure

We will try keeping two sub directories for fire and non fire images for training and validating . there are few data sets which default comes with this directory structure ,but for few dataset we are using running scripts to rearranged the names and locations to follow the above mentioned structure . this will basically makes the jobs easier ,although its not a necessary step .unless we are using image generator function in prepossessing library of kereas.

2.1.1 Getting Datasets

As currently there are lot of places where we can get the any required dataset .but for now we will use Kaggle availables datasets.

Chapter 3

PreRequisites/Models

3.1 PreRequisites

This work assumes some deep learning familiarity with topics like Neural Nets, CNNs, RNNs, Transfer Learning, Python programming and Keras library. The two below mentioned models will be used for our problem

3.1.1 Build Custom CNN Architecture

We are going to use TensorFlow API Keras for building our model.first create our ImageDataGenerator for labeling our data. datasets are used here for training. Finally, we will have 800 images for training and 200 images for validation. We are going to use data augmentation as well.

Here we have 3 data augmentation techniques which are applied

- horizontal flipping,
- rotation,
- height shifting.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_197 (Conv2D)         (None, 54, 54, 96)        34944

 max_pooling2d_17 (MaxPoolin  (None, 26, 26, 96)       0
 g2D)

 conv2d_198 (Conv2D)         (None, 22, 22, 256)       614656

 max_pooling2d_18 (MaxPoolin  (None, 10, 10, 256)      0
 g2D)

 conv2d_199 (Conv2D)         (None, 6, 6, 384)         2457984

 max_pooling2d_19 (MaxPoolin  (None, 2, 2, 384)        0
 g2D)

 flatten_3 (Flatten)         (None, 1536)              0

 dropout_13 (Dropout)        (None, 1536)              0

 dense_15 (Dense)            (None, 2048)              3147776

 dropout_14 (Dropout)        (None, 2048)              0

 dense_16 (Dense)            (None, 1024)              2098176

 dropout_15 (Dropout)        (None, 1024)              0

 dense_17 (Dense)            (None, 2)                 2050

=================================================================
Total params: 8,355,586
Trainable params: 8,355,586
Non-trainable params: 0
_________________________________________________________________
```

FIGURE 3.1: Model Summary

Now, we will create our CNN model. The model contains three Conv2D-MaxPooling2D layers pairs followed by 3 Dense layers. To overcome the problem of overfitting we will also add dropout layers. The last layer is the softmax layer which will give us the probability distribution for both the classes — Fire and Nonfire. One can also use 'sigmoid' activation function at the last layer by changing the number of classes to 1.

Here we are Using Adam Optimizer which involves a combination of two gradient descent methodologies:

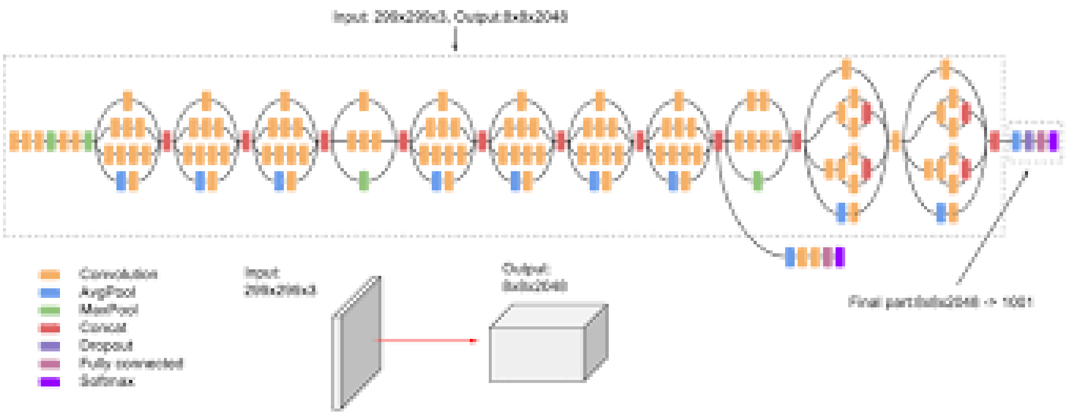


FIGURE 3.2: InceptionV3 Model

3.1.1.1 Momentum

This algorithm is used to accelerate the gradient descent algorithm by taking into consideration the ‘exponentially weighted average’ of the gradients. Using averages makes the algorithm converge towards the minima in a faster pace.

3.1.1.2 Root Mean Square Propagation (RMSP)

Root mean square prop or RMSprop is an adaptive learning algorithm that tries to improve AdaGrad. Instead of taking the cumulative sum of squared gradients like in AdaGrad, it takes the ‘exponential moving average’.

3.1.2 InceptionV3

We used Inception v3 which is a convolutional neural network for assisting in image analysis and object detection Shown in Fig [3.2]

3.1.2.1 InceptionV3

We can see that our above-created model is making a mistake in classifying my image. The model is 52% sure that the image has fire in it. This is because of the dataset it has

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

Figure 3.3: InceptionV3 Table

been trained on. There are very few images in the dataset that teaches a model about indoor fires.

So, the model only knows about outdoor fires and hence it errs when given an indoor fire-like shaded image. Another reason is that our model is not a complex one that can learn complex features of fire. SO next we use a standard InceptionV3 model and customize it. A complex model is capable of learning the complex features from the images.

We will use a different dataset this time, the one which contains outdoor as well as indoor fire images. I have trained our previous CNN model in this dataset and the result was that it overfitted, as it could not handle this comparatively larger dataset and learn complex features from the images.

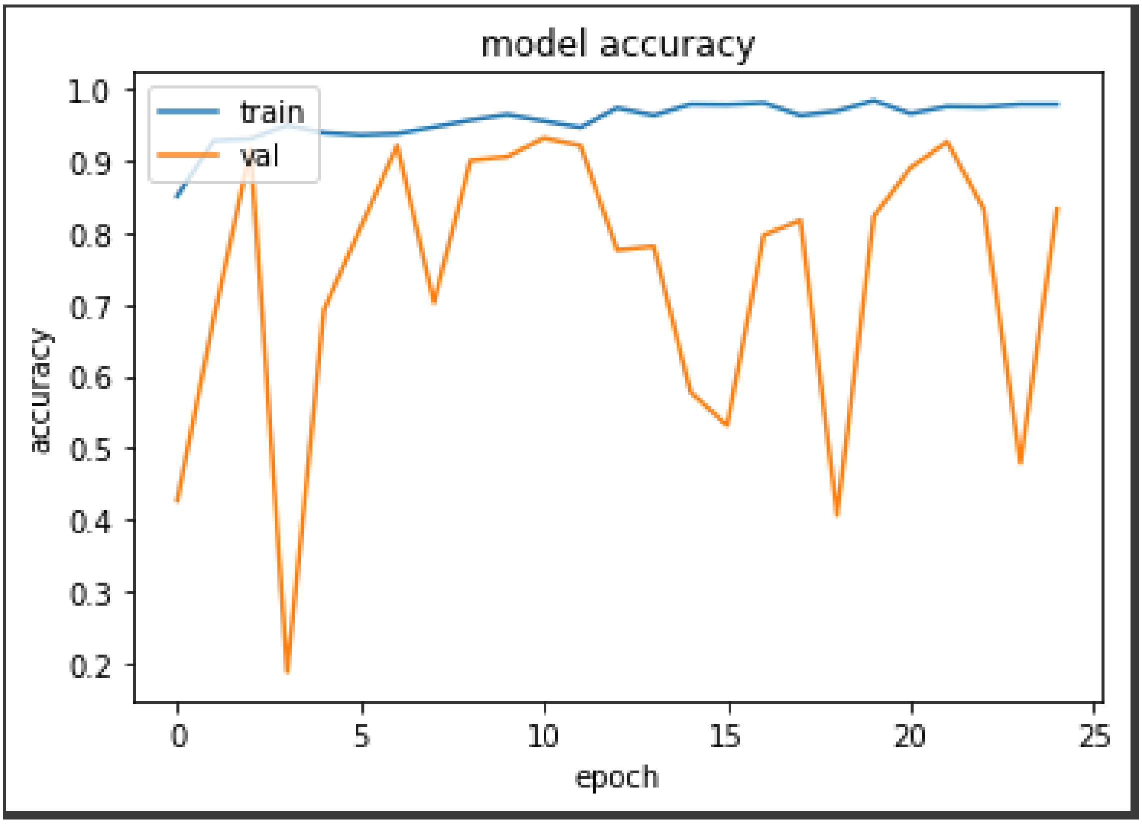


Figure 3.4: Epoch Vs Accuracy

Import our InceptionV3 model from the Keras API. We will add our layers at the top of the InceptionV3 model . We will add a global spatial average pooling layer followed by 2 dense layers and 2 dropout layers to ensure that our model does not overfit. At last, we will add a softmax activated dense layer for 2 classes.

Next, we will first train only the layers that we added and are randomly initialized. We will use RMSprop as an optimizer here.

After training our top layers for 20 epochs, we will freeze the first 249 layers of the models and train the rest i.e the top 2 inception blocks. Here, we will use SGD as an optimizer with a learning rate of 0.0001.

Here we Can see the results we are getting is more accurate

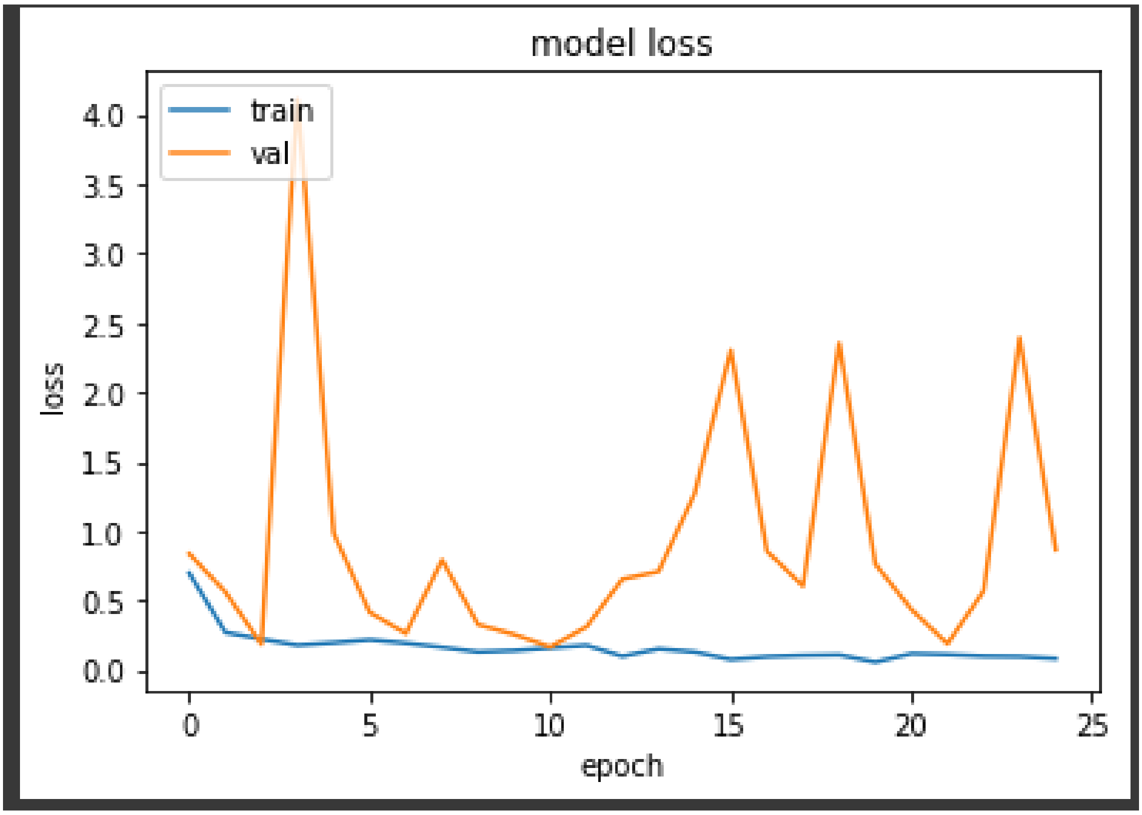


Figure 3.5: Epoch Vs Loss

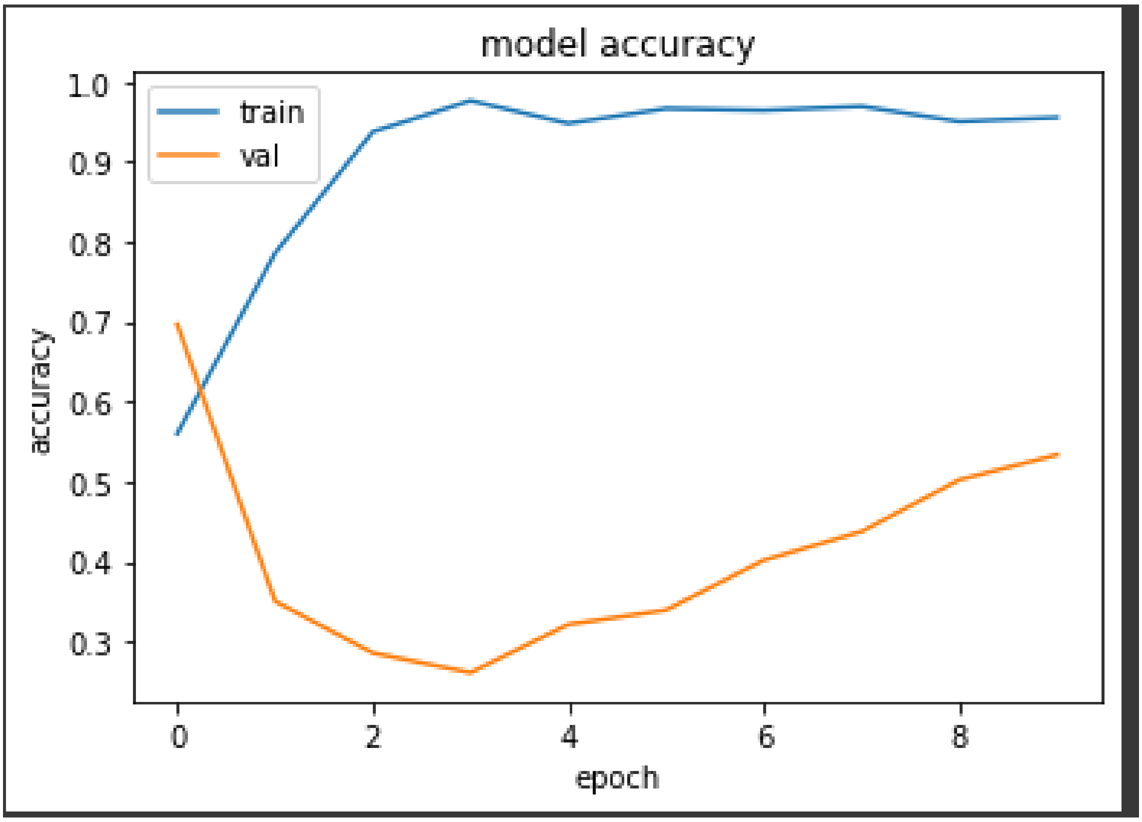


Figure 3.6: Epoch Vs Accuracy

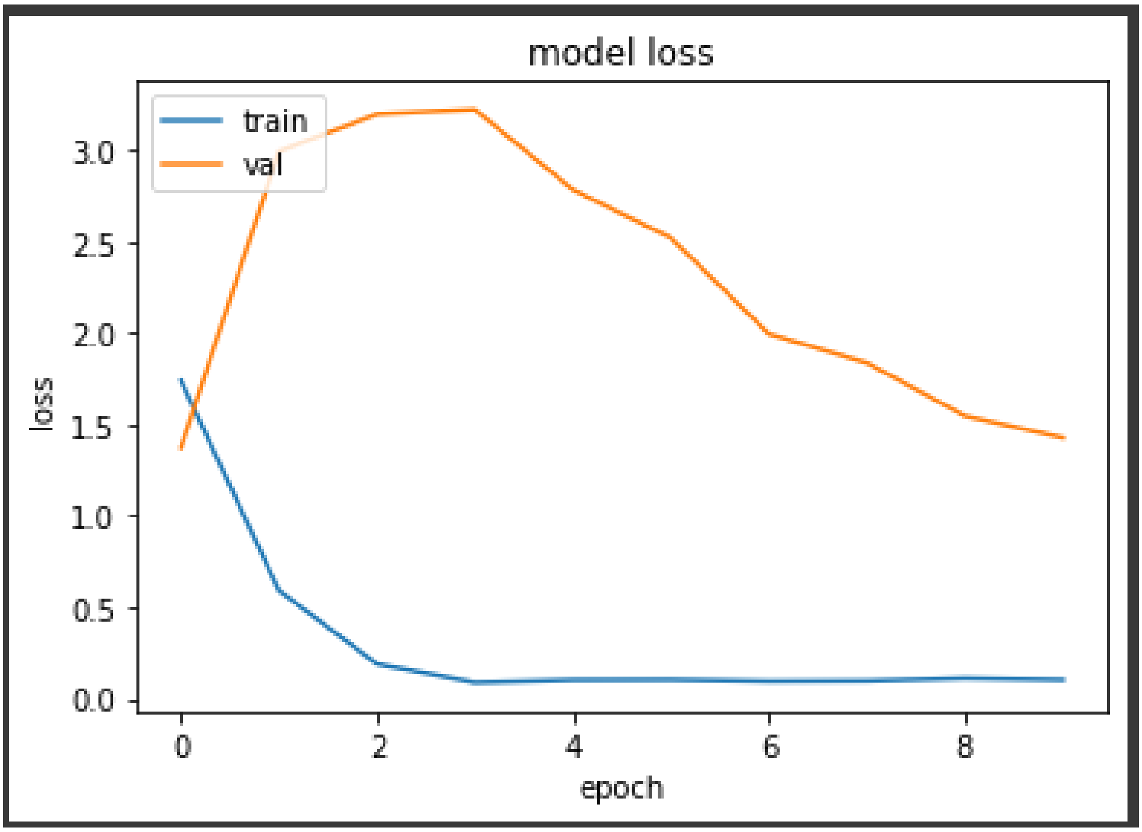


FIGURE 3.7: Epoch Vs Loss

Chapter 4

Conclusions and Learning's

Using smart cameras you can identify various suspicious incidents such as collisions, medical emergencies, and fires. Of such, fire is the most dangerous abnormal occurrence, because failure to control it at an early stage can lead to huge disasters, leading to human, ecological and economic losses. Inspired by the great potential of CNNs, we can detect fire from images or videos at an early stage. This report shows two custom models for fire detection. Considering the fair fire detection accuracy of the CNN model, it can be of assistance to disaster management teams in managing fire disasters on time, thus preventing huge losses.

Whats next can be done

There are lot of ways this can be implemented ,for low configuration system ,cloud could be the solution . it can be even modified to use for different categories of fire based hazards such as indoor fire ,outdoor fire ,smoke detection ,

References

[1] "https://arxiv.org/pdf/1409.4842.pdf."

[2] "https://arxiv.org/pdf/1312.4400.pdf."

[3] "https://keras.io/api/applications/."

[4] "https://www.kaggle.com/atulyakumar98/test-dataset."

[5] "https://www.kaggle.com/phylake1337/fire-dataset."

Fire Detection Using Deep Learning

Presented By

Arindam Sharma
CED17I022

Introduction

Problem Statement

There are lots of cases when surveillance shows fire hazards but nothing can you done as they are not equipped with the Fire Detection System ,there are only being alerted when fire alarm is active , there are currently lot of system which are equipped with visual bases system ,which can be used in general to stop fire hazards .

Here i will discuss about the CNN mode to Detect fire using any visual equipped system (e.g surveillance system) here i have used the   InceptionV3 Model ,compared with custom cnn model to detect fire in the frame

Motivation

Video Based System has advantages

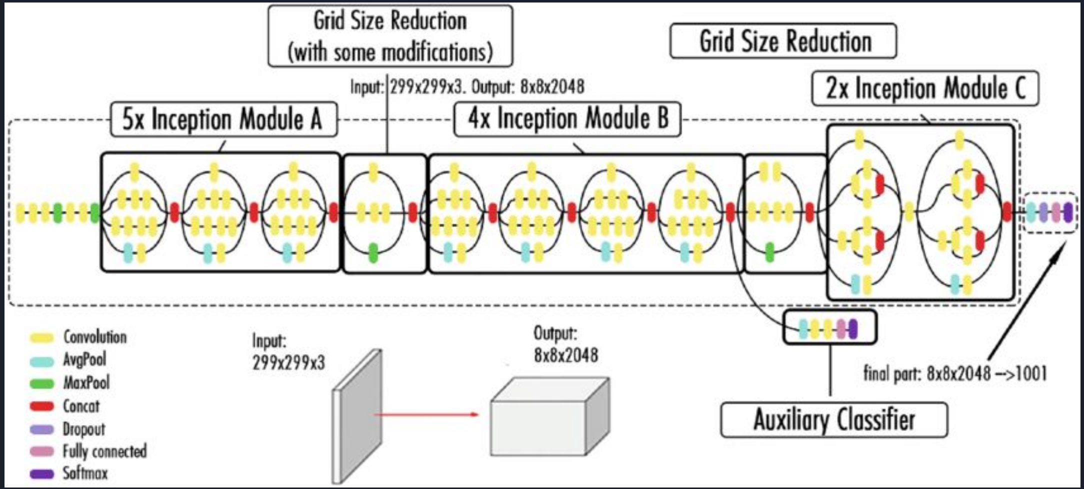
- Faster response on cheap cameras
- Safe ,more easy to use
- More effective in certain situations where fire detector are not useful.

Latency for smoke detectors are high with high ceiling buildings (places like auditorium, storage structure ,etc.)

Use of CNN architecture

Here we have worked on two CNN models

- Custom CNN Model
- InceptionV3 model

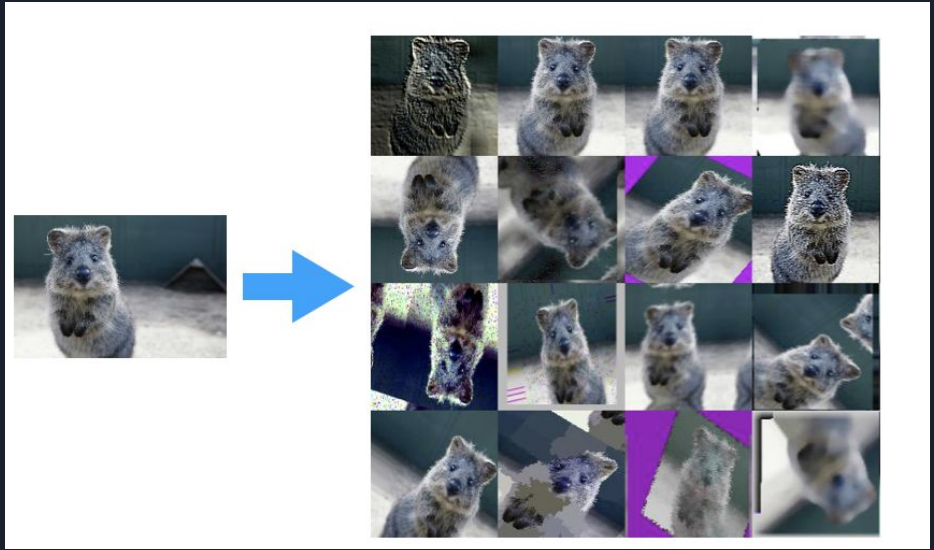


type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

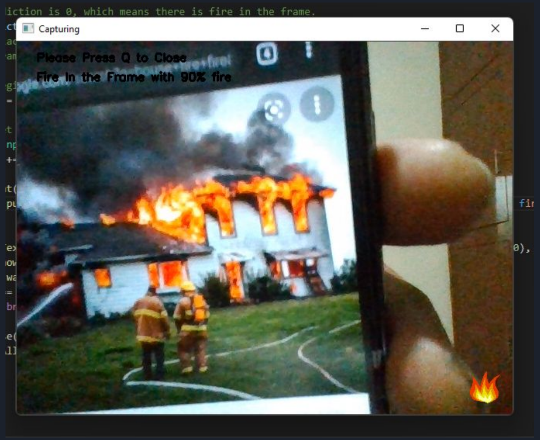
Data Augmentation

Variations

- Horizontal Flip
- Zoom
- Rotate
- Height Shifts



Demonstration



What next can be done

- Hosted in Cloud : Easily Used by low performing system.
- Can used for categorising fire types
    - Indoor fire
    - Smoke (initial stage of Fire)

Thank You