Flame Detection Using Deep Learning

Dongqing Shen

A report submitted to the Auckland University of Technology
in partial fulfillment of the requirements for the degree of
Master of Computer and Information Sciences (MCIS)

2017

School of Engineering, Computer, and Mathematical Sciences

Abstract

Video and images processing has been widely used in surveillance. Flame detection is one of the important issues in intelligent surveillance. In flame detection, we need to extract features. The color features were mostly used for flame detection and have been proved to be one efficient method. There are other features such as temperature and shapes. Many models based on these features were employed to the previous research, such as color model, fuzzy model, motion and shape model, etc.

Deep learning is also a novel method which could be much efficient and accurate in fire detection. In deep learning, the convolutional neural network (CNN) is one of the powerful methods for object detection. In this thesis, we are going to use the CNN to implement flame detection and compare it with those shallow learning methods so as to find out which method could be the most efficient one for flame detection.

Our contribution of this report is to use optimized YOLO model in deep learning for fire detection, the accuracy of flame detection is up to 81%.

Keywords: Flame detection; image processing; colour model; deep learning

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: Dongqing Shen Date: 23 June 2017

Acknowledgment

This research work was completed as the part of the Master of Computer and Information Sciences (MCIS) at the School of Engineering, Computer and Mathematical Sciences (SCMS) in the Faculty of Design and Creative Technologies (DCT) at the Auckland University of Technology (AUT) in New Zealand. I would like to deeply thank my parents for the financial support during my entire time of academic study in Auckland.

My deepest thanks are to my supervisor Dr. Wei Qi Yan who has provided me with much technological guidance and support. I believe that I could not have been able to achieve my Master Degree without his invaluable help and supervision.

Dongqing Shen

Auckland, New Zealand

June 2017

Chapter 1
Introduction

The first chapter of this report consists of five sections. In the first section, the background and motivation of flame detection will be introduced. Flame detection is widely used in fire alarming system; using image processing method could reduce human interference. This chapter also covers deep learning method, then research questions of how deep learning works in flame detection will be followed. The objectives will be discussed in Chapter 4. Finally, we will present an overview of the structure of this report.

1.1 Background and Motivation

Video and image processing assists in many aspects of surveillance problems: from object detection to face recognition using a digital camera which is efficient for saving resources and overwhelmingly improving the accuracy. Video and image processing is able to detect objects and human behaviors, identify human faces by using machine learning.

Flame detection using digital video and image processing could save a lot of manual work and have a higher accuracy. In decades, the fire detection was associated with smoking alarms which inspected fire by counting the density of little particles and smoke dust. The method was proved to be low accuracy accompanied with false alarms (Celik, 2010) and wasted a huge amount of resources, it also may miss the real fire incidents. Therefore, it is not suitable for fire detection happening in a large area such as forests.

Using Closed Circuit Television (CCTV) for fire detection could lessen security staff's human labor; hence, it is appropriate for fire detection in a massive area. Digital devices could briefly capture fire flames (Marbach, et al., 2006) and activate an incident management.

Feature model is a set of methods that allow a camera to fetch visual features of fire flames such as color model, motion model (Mathi & Latha, 2016; Vicente & Guillemant, 2002), shape detection (Toulouse, et al., 2016; Wang, et al., 2016), fuzzy pattern (Liu & Ahuja, 2004), hidden Markov model (Toreyin, et al., 2005), and so on.

Color model is based on the extraction of fire colors and could easily identify fire flames from digital images (Celik, Ozkaramanl & Demirel, 2007; Cho, et al., 2008; Vijaylaxmi & Sajjan, 2016; Hanamaraddi, 2016). The color of fire is not simple and it has different colors in burning area. Mostly there are three fire colors based on the temperatures which are at the flame center, a middle region, and the outside. The colors are changing corresponding to temperature variations within a flame. The highest temperature of a fire flame will show the color near white due to supplying with sufficient oxygen. It turns to orange when the temperature decreases which means from the core to the middle, and then it changes to red. Meanwhile, different burning materials will generate distinct colors, such as the fire of candle, coal, gas, or a match, etc. In a word, the color of fire is complicated, therefore, the color models are usually defined as a set of color patterns. The advantage of a color model is that it has a high accuracy of detection.

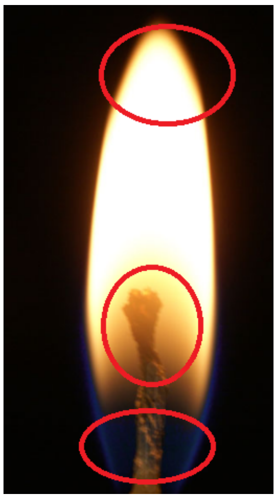


Figure 1.1.1 Different regions of a fire flame

Motion and shape are two visual features of a fire flame. Marbach (2006) proposed the fuzzy pattern for fire detection. In the pattern, the motion area will be set as the Region of Interest (ROI). A fire flame will be judged by fuzzy patterns. Edge detection of flames has also been thought as one possible solution (Kawa, Khartade, Sonawane & Madole, 2016). In the flame detection, the cost has to be taken into consideration (Toreyin, et al., 2006).

We think a single feature cannot improve the accuracy of fire detection but a combination of different features could increase the accuracy a lot (Jun, Yang, and Dong, 2009). Furthermore, the algorithm will be more fitful for by using a relative complete training set for deep learning.

The color is one of the most significant features of fire flame, most of the fire detection applications are based on color models. The RGB color model includes three basic color channels (red, green and blue) to identify the fire flames from a digital image.

The fire could easily be extracted from images by using colors. It may be due to illumination of light or bright noises from the images. All these situations could descend the accuracy of flame detection.

Machine learning is another efficient method for digital image processing. Deep learning is one of the latest concept based on artificial neural network (LeCun, Bengio & Hinton, 2015). With a training set, the learning process will be encapsulated in the black box to identify a fire flame which could increase the accuracy of fire detection. In addition, in deep learning, fire detection process is end-to-end, it does not require too much domain knowledge of a fire flame. Typically, CNN will be one of the most useful network models for object detection. It also has many frameworks for increasing performance of object detection and recognition.

1.2 Research Questions

As we mentioned, this project aims at finding out one efficient model of fire detection which will be implemented in three main methods: color model, combination RGB and HSI model, and deep neural network. Analyzing and evaluating the methods and techniques help us to find the most suitable method in this report. Therefore, there are three research questions in this report:

1. What methods in image processing could be applied to flame detection?

There are many methods that have been adapted for flame detection. Color model is one of them and could be the most popular one used so far. Also, deep learning has become one technique used in the field of digital image processing and the convolutional neural network is mostly used in the field of object detection and recognition.

2. How would these methods perform?

In this report, we will implement both shallow learning and deep learning methods to find the answer to the last question.

3. "Which methods would have a better performance for fire detection, shallow learning or deep learning?"

In flame detection, there are not many deep learning methods. In this report, we will find out the performance of deep learning working on flame detection and compare it with the shallow learning. By comparing the two method we will evaluate the advantage of using deep learning method.

1.3 Contributions

The focus of this report is on implementing both methods of shallow learning (using color model and the combination of color model and HSI model) and deep learning (using CNN). As most methods of flame detection are based on shallow learning, such as color model, fuzzy model, shape model; there are not so many deep learning methods which have been employed for flame detection. We will use optimized YOLO model to see how deep learning works in flame detection, and why we are going to use the deep learning method.

In addition, multiple deep learning methods will be discussed in this report. There are many CNN frameworks which have been proposed recently, we do not have time to implement all of them but we will evaluate them in object detection. Last but not the least, three detection methods will be presented and compared with each other.

Our neural network model is based on YOLO framework. The reason we used YOLO is to decrease time cost of training. In the training set, we used 172 flame images as our training set and to increase the set of our training data, we used data enhancement to increase the training data to 1720.

We used flipping, changing the brightness and lightness to increase the amount of training data.

With the feature extraction, the advantage of the CNN is that the feature maps are created by the networks. Unlike the shallow learning (color model), the feature models are created by us and using the exist feature model to detect the object. While the CNN can extract the feature by the neural network itself.

For classification part, because we only need one object detected which is fire, so the classification problem would be flame or not flame.

In the training phase, as we mentioned that we used 1720 training data set and we used 2 epochs as pre-training and 4 epochs as formal-training. The whole training process cost 90 minutes and the accuracy of the detection reaches 81%.

1.4 Objectives of This Report

Firstly, we will introduce image processing techniques used in flame detection. Some of these methods will be demonstrated in this report, especially color based model and CNN framework.

Secondly, we will implement both shallow learning and deep learning methods. There are many shallow learning methods used for fire detection, such as color model, fuzzy model, shape model, and motion model, etc. Color based model is the method most used and also has excellent performance in accuracy. So, we will implement two color models as the shallow learning. For deep learning, we will use YOLO to find out how deep learning method performs in flame detection. The original YOLO has 24 layers. Therefore, we will optimize it because we only have one object to detect in lieu of multiple objects. Finally, we are going to compare the performance of two main methods for fire detection. The accuracy is the most important aspect of evaluation of flame detection.

1.5 Structure of This Report

The report is structured as follows. In Chapter 2, literature will be reviewed, such as the previous studies in flame detection and the basic research of convolution neural network. This chapter will also demonstrate the color based models for flame detection.

In Chapter 3, the reason why we choose YOLO as the framework will be explained. Furthermore, we will present two color models, the design of this simplified YOLO model in deep learning will be illustrated.

In Chapter 4, all proposed methods will be demonstrated. For deep learning method, the accuracy of flame detection will be emphasized.

In Chapter 5, analysis and discussions based on the results of Chapter 4 will be detailed. Meanwhile, the comparison of all methods will be presented in Chapter 5. Finally, the conclusion and future work will be presented in Chapter 6. The possible optimized methods will be probed in this chapter.

Chapter 2
Literature Review

With in-depth analysis of the research questions and rationale reviews of the previous studies, the focus of this report is on flame detection from digital images. Most of the methods used are shallow learning, for instance, color model, motion model, and shape model, etc. Meanwhile, deep learning methods are seldom used in flame detection.

2.1 Color Model for Fire Detection

2.1.1 HSI Model

HSI color space has been employed for increasing the accuracy of flame detection. In HIS, H stands for the component hue of a color space, S means saturation, and I refers to color intensity. The establishment of HSI color space is based on two facts: the channel I could avoid color effect; the channels H and S strongly relate to human sense. These characteristics make HSI space being a suitable model for flame detection and analysis. Also, HIS color space and RGB space could be converted mutually.

2.1.2 Combination of RGB and HIS Spaces

RGB color space could efficiently extract the flame region of an image in a very short time; meanwhile, the HSI space provides a more accurate method of feature extraction. One new method will be proposed in this report which is to combine both color spaces together to create one more robust model.

2.1.3 YCbCr Model

YCbCr is also a color space. Y is for luminance, Cb and Cr are for blue and red chroma components. And also, YCbCr color space could convert to RGB with little precision reduction (YCbCr is 24 bits data while RGB is 16 bits). Yang, Yuhua, and Zhaoguang (2007) have presented one algorithm for converting the two color spaces. Celik and Demirel (2009) have used YCbCr space to replace RGB for detecting fire flames from acquired images, the results show higher detection rates (>99%) and lower false alarm rates.

2.2 Other Feature Model Used for Flame Detection

Except for color model, there are many other feature models used for fire detection, such as motion model, fuzzy model, shape model, etc. Multiple feature model will be used to increase the accuracy of fire detection. Qi and Ebert (2009) demonstrate one method not only uses color model and motion model but also takes the temporal variation of flame intensity into consideration. The method combined three color models (Celik, et al., 2007) could have a very high performance in accuracy (> 98.2%). The novel method used color features and back-propagation neural network to classify smoke, and calculated motion feature by using optical flow (Chunyu, et al., 2010). Ko, Cheong and Nam (2009) added support vector machine (SVM) to the methods and used it to classify fire pixel variations which make the result more robust to noise.

2.3 Deep learning

Deep learning allows computational models to learn data with multiple layers of abstraction. Deep learning is based on artificial neural network which has been used in image and video processing. Decades ago, ANN methods were very popularly used in image and video processing, such as back propagation algorithm (Mozer, 1989) and logistic regressions.

Comparing to the shallow learning methods (such as BP algorithm, Hecht-Nielsen, 1988), deep learning proposed a network with deeper layers. With the development of ANNs, deep learning has become a very efficient way to solve the problems from image and video processing, and computer vision (Wan, et al., 2014). It has been applied to many research fields such as face recognition, image classification (Chan, et al. 2015) and object detection. As same as other methods of object detection, the methods of flame detection could benefit from deep learning which may even more efficient compared to the color based models.

2.4 Deep Learning Networks

Deep learning is one branch of machine learning and can be also considered as the development of artificial neural networks (ANNs). There are many methods proposed for digital image processing, such as support vector machines (SVM) and logistic regression

(LR). Some of these methods have already been applied to flame detection (Kong, et al., 2016).

The layers of traditional neural networks have been used in deep learning which includes input layer, hidden layer, and an output layer.

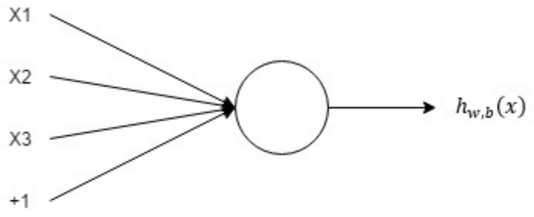


Figure 2.4.1 Single neural unit in the neural network

$$h_{w,b}(x) = f(W^T x) = f(\textstyle\sum_{i=1}^{3} W_i x_i + b) \tag{2.1}$$

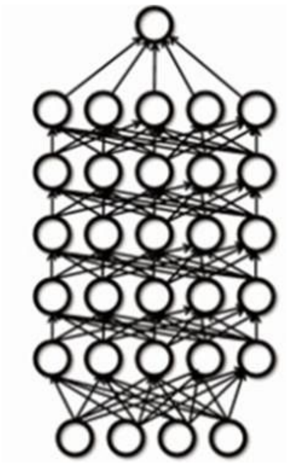


Figure 1.4.2 The deep learning model with multiple hidden layers

To overcome the problems in the tradition neural network, the deep learning adopts one different training method (Schmidhuber, 2015). The iteration algorithm is employed to train the network by using back propagation in traditional neural networks. With initial values, the current network's output is calculated, the parameters are adjusted from the former layer through the changes between the current value and the label.

Using back propagation may cause the issue of gradient diffusion in a deep network which usually has seven or more layers. Deep learning adopts a layer-wise mechanism to solve this problem.

2.5 Convolutional Neural Network

Convolutional neural network is one of the artificial neural networks and has become one of most efficient methods for image classification (Zeng, et al., 2014; Ciresan, et al., 2011; Kim, 2014), speech analyzing (Lecun & Bengio, 1995; Kalchbrenner, Grefenstette & Blunsom, 2014;), and object recognition (Lawrence, Giles, Tsoi & Back, 1997). The weights sharing are more like a biological neural network which decreases the complexity of a network model and an amount of weight values.

The convolutional layer might be more in the application (Krizhevsky, Sutskever & Hinton, 2012; Matsugu, Mori, Mitari & Kaneda, 2003). The purpose of multiple convolutions is to ensure that the learned features are more global. Because one convolution may only extract partial features, which may decrease the accuracy of flame detection.

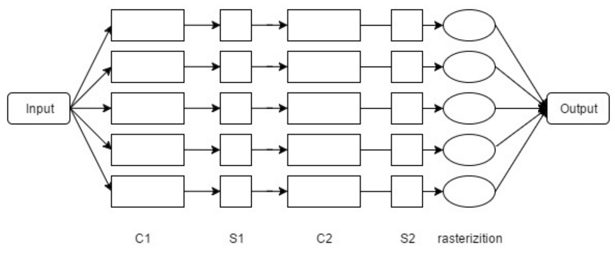


Figure 2.5.1 Basic model of CNN

CNN is one network with multiple layers that is consisting of several two-dimensional flat and each flat is composed by unique neuron. The input image will be convoluted into several filters to form the first convolutional layer (Displayed as C1 in Fig.2.5.1). Smith and Topin (2016) proposed 14 designed patterns for the convolutional neural network. By weights, bias and summating every four pixels in an image, we can obtain feature maps through an activation function (tanh function, sigmoid function, ReLu). It could also use hyperbolic tangent. In this report, the Leaky ReLu will be used as our activate function because it could have a better performance (Dal, Sainath & Hinton, 2013).

$$f(x) = \tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}\ , f(x) = |\tanh(x)| \tag{2.2}$$

$$f(x) = (1 + e^{-x})^{-1} \tag{2.3}$$

$$f(x) = max(0, x) \tag{2.4}$$

All the feature maps will be filtered again to obtain C$_2$, as same as the former step to get S$_2$. At last, all the pixels will be rasterized as the traditional neural networks to acquire the results (Sun, Wang & Tang, 2014).

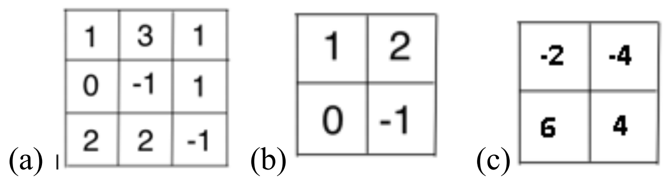


Figure 2.5.2 Convolution operations

Figure 2.5.2 shows the operation of convolution, (a) shows the input and (b) is the filter. To conduct convolution operation, the first step should be flip the filter and using the flipped filter matrix to multiply the input so that we can get the output which is c.

In image processing, the image will be presented as a matrix includes vectors which stand for pixels' information. If there is a picture, its size is 1000×1000, it could be presented as a 10^6 vector. Meanwhile, it could have 10^6 hidden neural units, which create 10^{12} weights for this image.

CNN provides one efficient approach for decreasing the depth of the network. First, the connectivity relates locally with space. As a 1000×1000 image, the neural units do not need to include the whole image, which only contains the local information of the picture. If the filter (Also known as kernel) is set as 10×10 (Fig.2.5.3), then it will only need to connect to this 10×10 local image which will decrease connectivity to10^8.

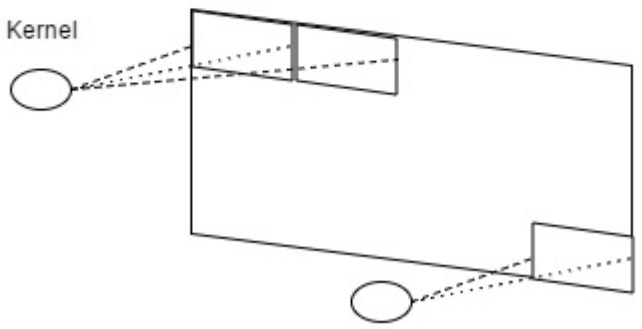


Figure 2.5.3 Convolutional net with the setting filters

In CNN, all these neural units will share the same parameterization (weight vector and bias) and form a feature map, which means all neurons are detected in the same feature in one convolutional layer. In a 1000×1000 image with the filter set as 10×10 (Fig.2.5.3), one neuron will have 100 parameters, and these parameters are all the same, then no matter how many hidden layers we have, there will only 100 parameters for the connectivity between two layers. However, this kind of feature extraction could decrease the precision. To solve this problem, CNN could also set up multiple filters. If there are 100 filters for the 1000×1000 image, there will be 10^4 parameters.

In applications, 3×3 (Simonyan & Zisserman, 2014) and 5×5 are mostly used as the size of the convolutional kernel. In a real operation, we may add padding to maintain the image size. For example, we may add one padding for the 3×3 kernel and two paddings for the 5×5 kernel.

The other important layer of CNN is pooling. Pooling layer is for extracting features by using down-sampling to simplify the complexity of the neural network. The errors of pooling are mostly based on two aspects:

(a) The limitation of the neighborhoods increases variance;

(b) The deviation of convolutional layers creates average bias.

Average Pooling can reduce the first bias; max Pooling (Giusti, et al., 2013) could reduce the latter one. There are other pooling methods used for the CNN, such as fractional max pooling (Graham, 2014), overlapping pooling (Krizhevsky, Sutskever & Hinton, 2012) and spatial pyramid pooling (He, et al., 2014).

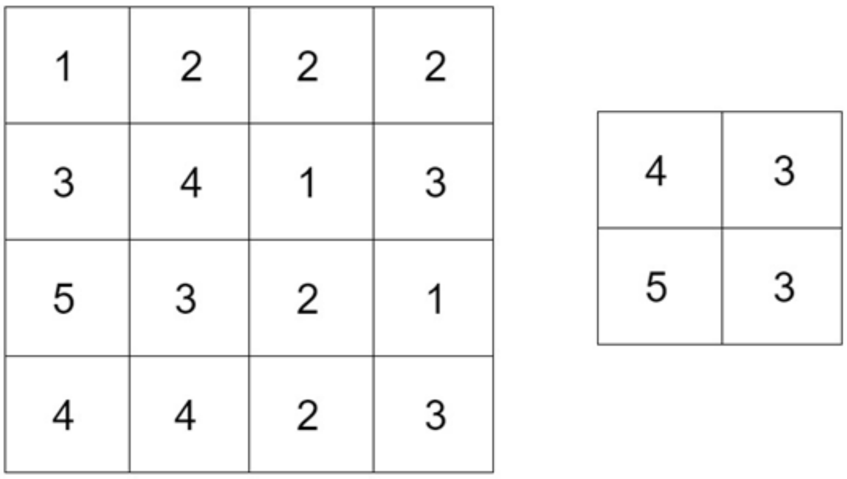


Figure 2.5.4 Max Pooling

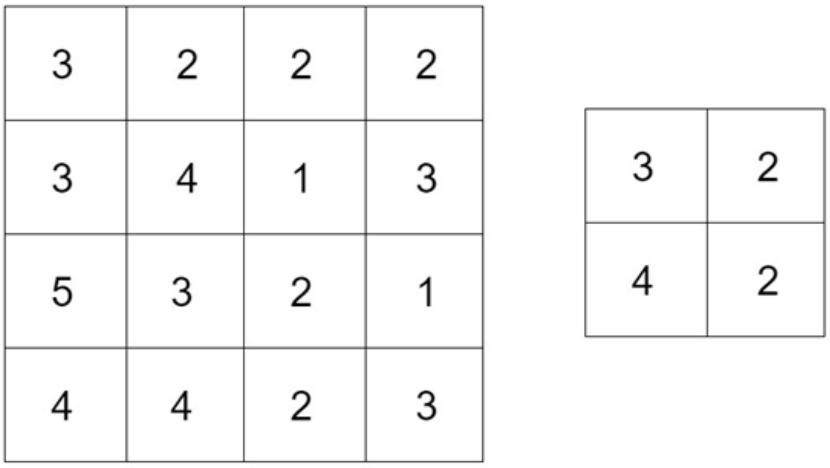


Figure 2.5.5 Average Pooling

Chapter 3
Methodology

The main content of this chapter is to introduce the method we choose. The chapter mainly covers the details of our research methodology, such as how to limit threshold of the color based model and how to optimize our CNN model.

3.1 Shallow Learning Design

We will use three color based models to extract fire flame from given images which include RGB based models, HSI based model and the combination of these two models. HSI color space is related to the RGB based model and the data from two models could be converted.

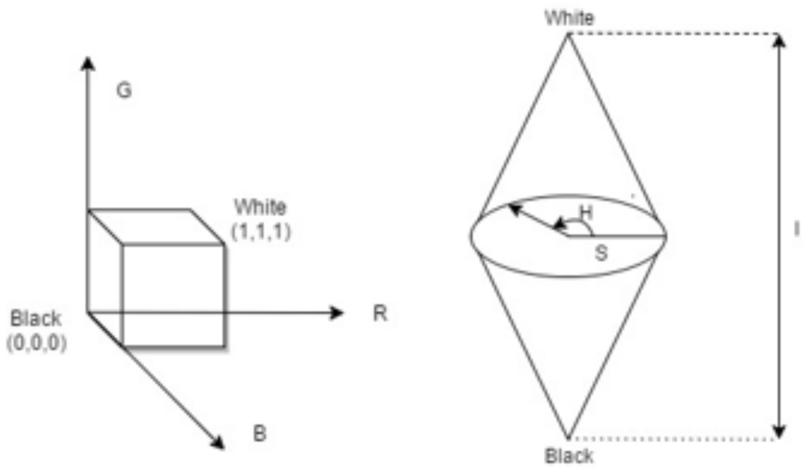


Figure 3.1.1 Color space conversion between RGB space and HIS space

The most common feature of fire flames is color. For most of the fire flames, the color is red, which means the R component of RGB space takes more effect than others. Usually, the green channel takes more import role than the blue channel. In the meantime, the red and green channels will be very large above the average. The criterion of flame detection is present as

$$R > R_{avg} \quad AND \quad G > G_{avg} \quad AND \quad R \geq G \geq B \tag{3.1}$$

In the Fig.3.1.1, the HSI data could be converted to the RGB space.

$$I = \frac{R+G+B}{3} \tag{3.2}$$

$$S = 1 - \frac{3}{(R+G+B)} \min(R, G, B) \tag{3.3}$$

$$H = \cos^{-1}\left(\frac{\frac{1}{2}(R-G)+(R-B)}{(R-G)^2+(R-B)(G-B)}\right)^{\frac{1}{2}} \tag{3.4}$$

where the values of H, S and I could be obtained from the RGB space. There are three threshold sets; it returns true if the three components all meet the condition of the threshold, otherwise, return false.

The condition is presented as

$$H_{min} < H < H_{max} \tag{3.5}$$

$$S_{min} < S < S_{max} \tag{3.6}$$

$$I_{min} < I < I_{max} \tag{3.7}$$

By combined RGB based model and HSI based model together, we will keep the two constraint conditions and add three new dependencies into the HSI model.

$$R > R_{avg} \quad AND \quad R \geq G \geq B \tag{3.8}$$

$$S > 0.2 \quad AND \quad S > \frac{(255-R)}{20} \quad AND \quad S \geq \frac{(255-R)\times ST}{RT} \tag{3.9}$$

where ST is the saturation threshold which will be set between 55 to 65 (Chen, Wu &Chiou, 2004). RT is the threshold for the red channel which will be set in the range of 115 to 135.

3.2 Design of Deep Learning Method

There are many deep learning architectures available such as Caffe, TensorFlow, MXNet, Torch, and Theano. Caffe is well known and widely used in machine vision with remarkable implementations of the convolutional neural network (Jia, et al., 2014). It has lots of extensions and is good for the existing network. However, it is not suitable for recurrent neural network and is also too ponderous for big networks such as GoogleNet and Residual Net.

Google creates TensorFlow to replace Theano. Like most deep learning framework, TensorFlow (Abadi, et al., 2016) is written with a Python API instead of C/C++ which makes it run faster. As same as Theano, it generates computational graphs and performs automatic differentiation, supports faster development. The disadvantages of TensorFlow is that it does not have many pre-trained models and occupies much more RAM (Comparing to Torch).

Torch is used in Facebook Research which makes it very famous in the world (Collobert, Bengio & Mariethoz, 2002). Torch optimizes the computation units so it is easy to write our own layer types and run on GPU. Torch also comprehensively supports all convolution operation, such as time convolution (useful for natural language processing) and 3D convolution (useful for video processing). The biggest limitation of Torch is the programming language. Torch does not have the interface of Python. It uses the programming language Lua which is inconvenient.

MXNet is adopted by Amazon Web Service which supports lots of programming languages (C++, Python, R, Julia). It has faster and flexible library, detailed document (Chen, et al., 2015). MXNet emphasizes the efficiency of RAM usage.

Theano (Bergstra, et al., 2010) is one of the most stable libraries. It allows automatic function gradient computations with Python interface which is integrated with Nummy. However, it does not have multi-GPU support nor horizontal capabilities, especially, when comparing to TensorFlow, Theano is to be left behind.

Considering the TensorFlow is one of the most popular frameworks and friendly for users, in this report, we will use TensorFlow as the solution for flame detection.

Table 3.2.1 Comparison of different deep learning framework

Library	**Caffe**	**TensorFlow**	**Torch**	**MXNet**	**Theano**
Language	C++/CUDA	C++/CUDA/Python	C/Lua/CUDA	C++/CUDA	C++/CUDA/Python
Hardware	CPU/GPU	CPU/GPU/Mobile	CPU/GPU/FPGA	CPU/GPU/Mobile	CPU/GPU
Speed	Fast	Medium	Fast	Fast	Medium
Flexibility	General	Excellent	Excellent	Excellent	Excellent
Document	Comprehensive	Medium	Comprehensive	Comprehensive	Medium
Suitable Model	CNN	CNN/RNN	CNN/RNN	CNN	CNN/RNN

3.3 Possible Methods of CNN: From RCNN to YOLO

There are many practical methods of CNN that could be used such as RCNN, SPP, Fast-RCNN, Faster-RCNN, and YOLO.

3.3.1 RCNN (Regions + CNN)

RCNN uses four steps to implement the object detection:

(a) Use SS (Selective Search) to get 1000-2000 proposals, resize them to match CNN input.

(b) Train every proposal to extract features.

(c) Classify object using trained SVM for each class.

(d) Train and use linear regression to adjust bounding box for the feature proposal.

The biggest problem of RCNN is that the training time and testing time are very long because it needs to get 1000-2000 proposals first and save them to disk, also these proposals need to be calculated in all the former layers which need lots of repetitions. In addition, fully connected layer is expected that all the vectors will have the same size, so all the proposals need to be resized using crop or wrap, both strategies are not suitable because the crop may cause the proposals are not fully extracted and the wrap could change scales of objects.

3.3.2 Fast RCNN

Fast RCNN was proposed by Girshick in 2015 which overcomes several problems of RCNN. The reason why we select RCNN is that it decreases the consumptions of time and space.

Fast RCNN still uses SS to get all region proposals but these proposals will directly send to the ROI Pooling layers, which decrease the repetitive calculation in the former feature layers. Also in the RoI Pooling layers, it resizes all data to a fixed size which is

6×6 to fit the FC layer.

What Fast RCNN has done is to replace RoI Pooling layer to the Pooling layer 5, and use Softmax to classify instead of SVM (Figure). The Softmax is one extension of logistic regression to the multi-classify problem.

$$f(z_j) = \frac{e^{z_j}}{\sum_{i=1}^{n} e^{z_j}} \tag{3.10}$$

The loss function of Softmax is:

$$loss = \log \sum_{i=0}^{n} e^{z_j} - z_j \tag{3.11}$$

The difference between Softmax and SVM is that SVM only gets the output object but Softmax can get the probability of each class.

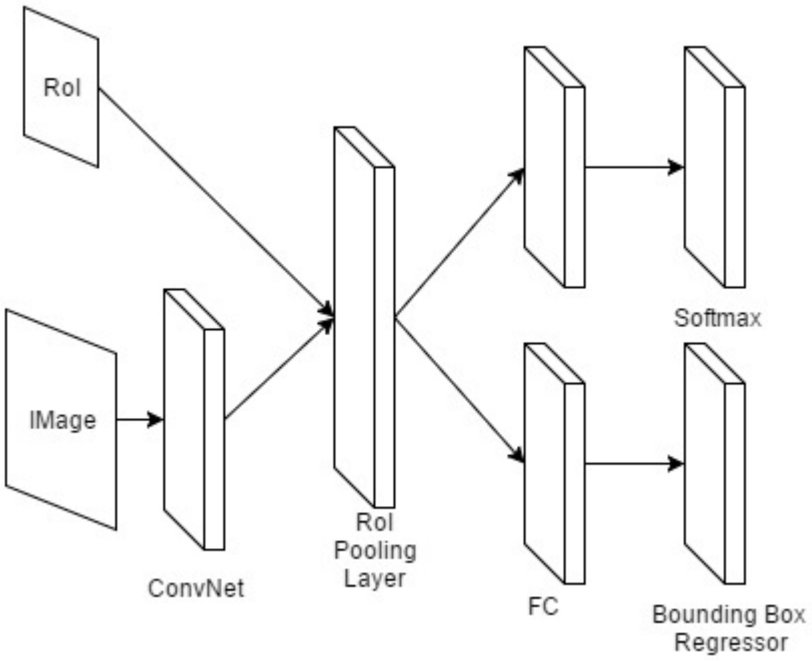


Figure 3.3.1 Fast RCNN: RoI Pooling layer

Fast RCNN Substantially decreases the training and testing time comparing to RCNN, the testing time per image decreases to 2 seconds from 50 seconds.

3.3.3 Faster RCNN

After Fast RCNN, Girshick proposed Faster RCNN to improve the training speed of the Fast RCNN. From RCNN to Faster RCNN, the four steps of object detection are finally unified into one network. Faster RCNN does not use selective search to get region proposals. Instead, it uses a region proposal network to carry out the same task. There is no repetition and all the calculations are performed by using GPUs.

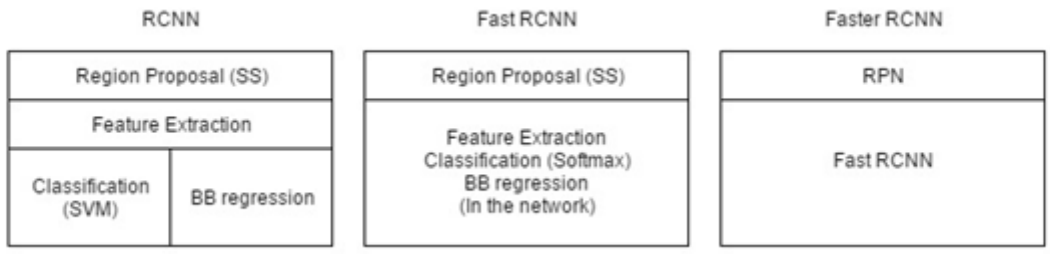


Figure 3.3.2 Comparison of RCNN, Fast RCNN, and Faster RCNN

RPN uses a 3×3 window sliding it on the feature map. Each sliding will create one bounding box, return the position of the box and classify these boxes are objects or not.

Table 3.3.1 Result of Faster RCNN

	RCNN	**Fast RCNN**	**Faster RCNN**
Time per image	50 seconds	2 seconds	0.2 seconds
mAP (VOC 2007)	66.0	66.9	66.9

3.3.4 YOLO

YOLO is one of the fast object detectors but it also sacrifices a little bit precision. Unlike other methods, YOLO did not create 2000 region proposals, it creates a $S \times S$ grid cells, each cell will be responsible for the object which falls into the cell. Every cell will predict the bounding box and the confidence score of this box. In Joseph (2015), for evaluating YOLO on VOC, a 7×7 bounding box and 20 labelled classes are defined, which means it only extracts 98 proposals. YOLO is faster than RCNN which needs 2000 proposals.

Although each cell could predict B bounding boxes, only the object with the highest IOU will be detected as output. So, there is only one output for each cell even there are multiple objects in one cell.

The error analysis of Fast RCNN and YOLO by using VOC 2007 is given as Fig.3.3.3.

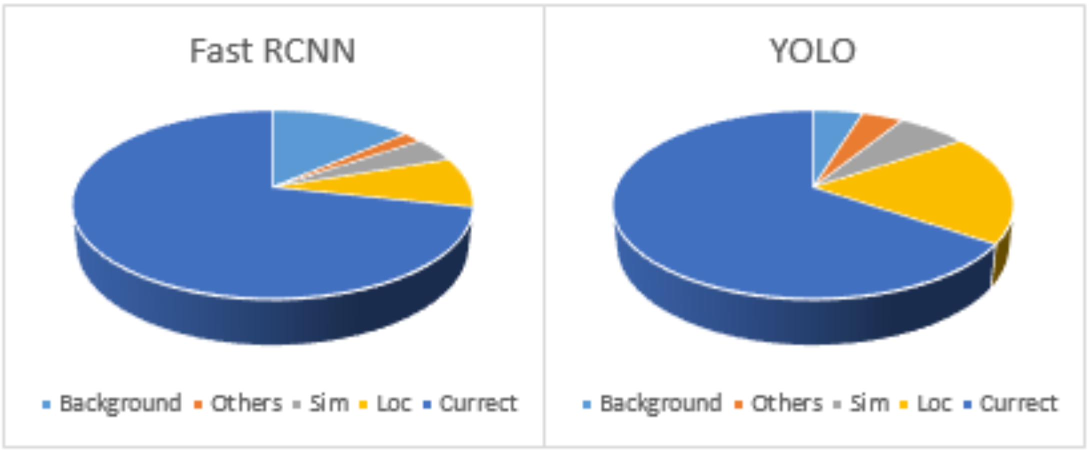


Figure 3.3.3 The error Analysis of Fast RCNN and YOLO (Redmon, et al., 2016)

Figure 3.3.3 demonstrates the error types in 20 classes. YOLO has a large amount of localization errors, even more than all other error types. The accuracy of YOLO is a bit lower than Fast RCNN, but it has a superior performance on background extraction.

YOLO used the whole image instead of a region proposal to train and testing which has a lower rate of background error. When comparing detection speed to another real-time system on PASCAL VOC 2007, YOLO has an overwhelming advantage. The fast RCNN takes around 2 seconds per image to generate bounding box proposals. The Faster RCNN as the most accurate model reaches 7 fps while a smaller model, which has lower mPA of 62.1 (Faster RCNN ZF), can achieve 18 fps. But YOLO could reach 45 fps, which is twice faster than RCNN ZF and even has a higher mPA value of 63.4.

The limitation of YOLO is that even each cell could predict B bounding boxes but there is only one class that could be detected, which makes small objects hardly to be detected.

3.4 Choose from CNN methods

Flame detection would rather have a higher speed detection, which makes YOLO as a good choice. The biggest interfering element in flame detection could be the background. Considering that a fire flame is not a small object in the whole image, which makes YOLO may have a better performance of accuracy. Therefore, YOLO may be one of the better choices for flame detection.

3.5 Optimize YOLO model for fire detection

The original YOLO uses 24 convolution layers with 2 fully connected layers. And there is one fast version of YOLO (Fast YOLO) which has fewer layers (nine) with the same training and testing parameters as YOLO. In flame detection, there is only one class needed to be detected. Hence, the classification would be fire or not fire instead of several different objects.

YOLO is better on detecting objects but does not have an excellent performance on localization, which might cause the problem on catching the flame position.

Meanwhile, for the single fire-flame detection, we expect to simplify YOLO's convolutional neural network. We used a model pretty like the Fast YOLO network model but we do not set a fully connected layer at last because of the limitations of graphics processing cards. Fortunately, we still have a good result. We use nine convolution layers and one more layer for pre-training; without a fully connected layer, we directly add the network for finding out the four parameters (x, y, width, and height). Each convolution layer will be followed by the max pooling layer and one activation function: ReLu.

For the loss function, because we have only one object needed to be detected, so there are only two loss functions: (a) Object/ No object classification, (b) Regression of bounding box (x, y, width, height).

3.6 Network Model

The original YOLO convolutional neural network has 24 convolutional layers, which it is more powerful to detect multiple objects. In this report, there is only one object to be detected, which means we do not need the same complex model as the original model of YOLO. In our model, we design a network having 12 convolutional layers. Figure 3.6.2 shows the constructor of every single convolutional layer.

First of all, we design nine pre-training convolutional network for positioning which grid has fire. The conv6 in the Figure. 3.6.1 has four convolutional layers.

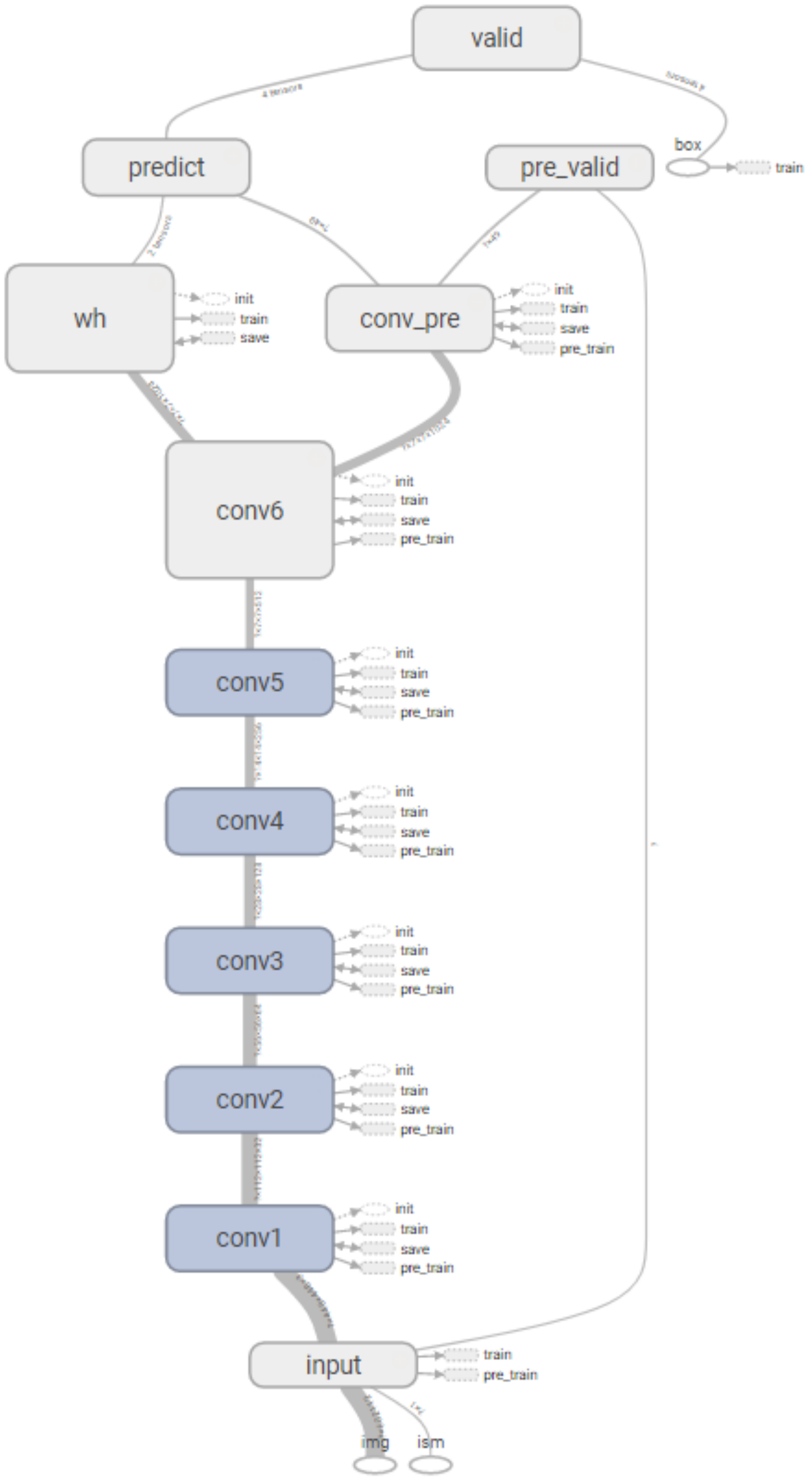


Figure 3.6.1 The optimized YOLO model for fire detection

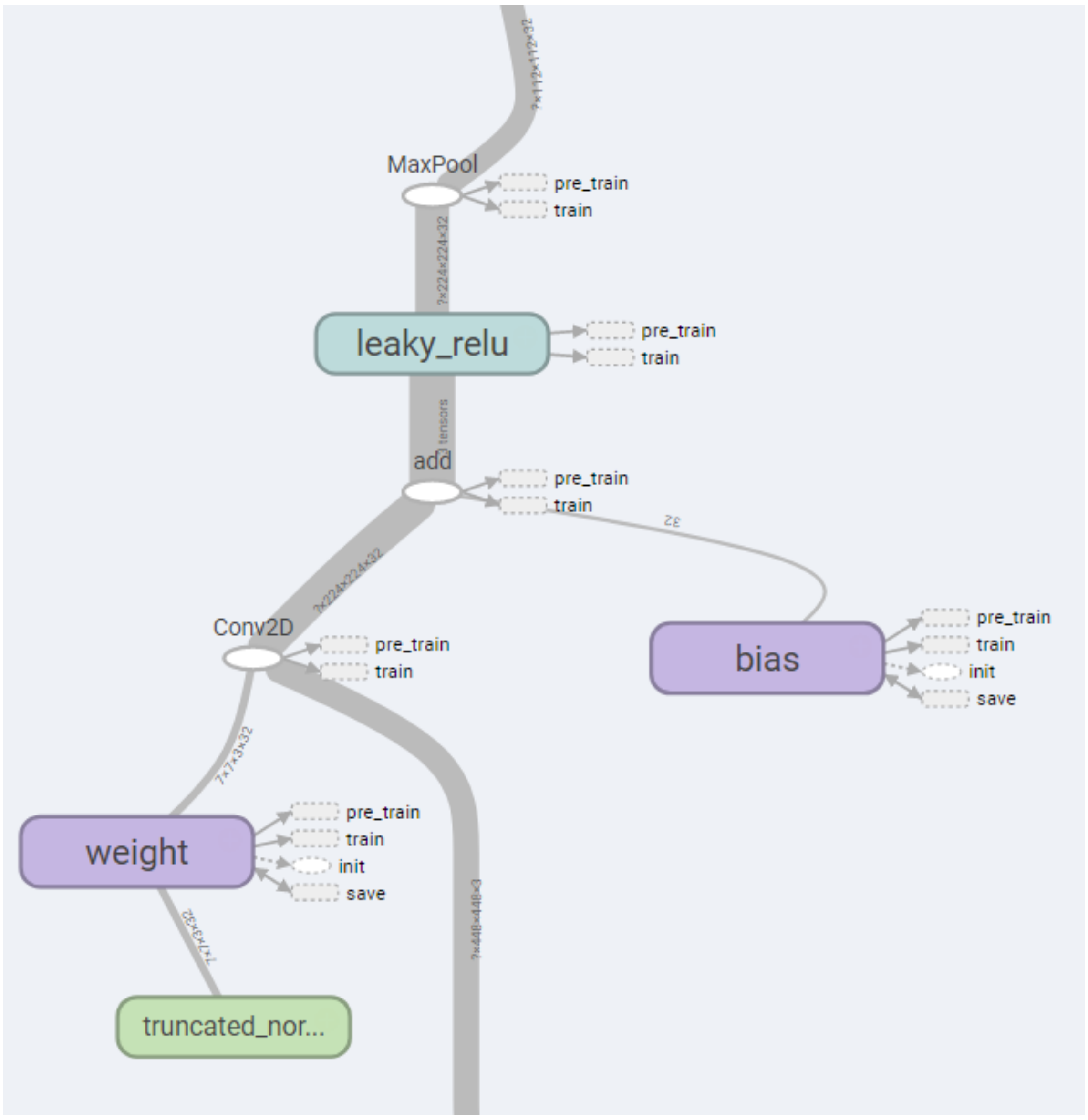


Figure 3.6.2 The first convolutional layer as a sample for every single convolutional layer

After pre-training, the formal training will be done in the section shown in Figure 3.6.3. The $7 \times 7 \times 1024$ feature matrix from con6 will be calculated after two more convolutional layers are set to a $7 \times 7 \times 516$ feature matrix. The $7 \times 7 \times 516$ feature matrix has been converted to 2D vector through two fully connected layers. Formal training is used to increase the accuracy of prediction of the bounding box.

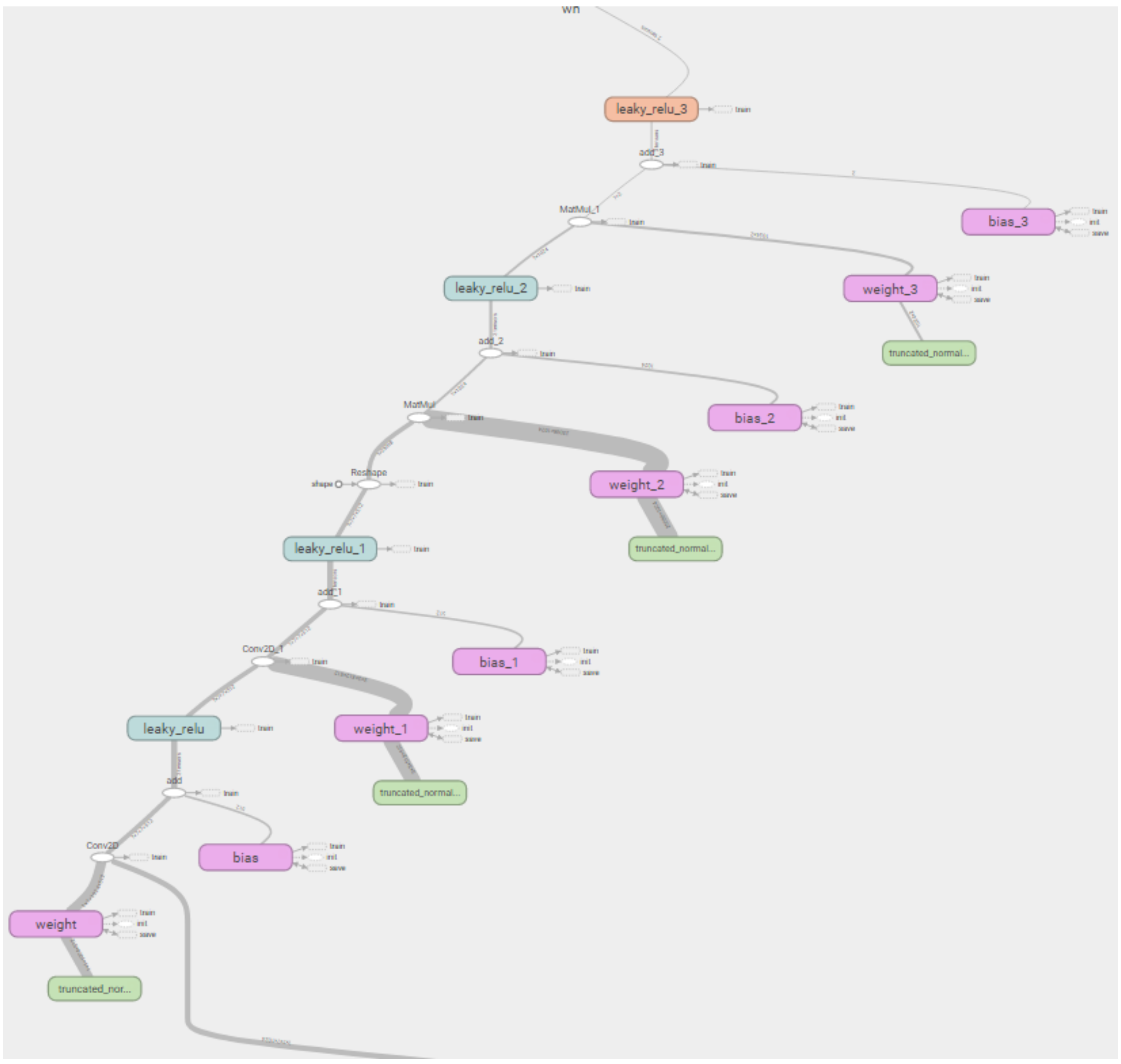


Figure 3.6.3 Formal-training layer which has two convolutional layers and two fully connected layers

Assume the predicted bounding box as S, the correct bounding box is as $\hat{s}$, the intersection as $S_{\cap}$, thus we have,

$$IoU = \frac{S_{\cap}}{S + \hat{s} - S_{\cap}} \tag{3.12}$$

Basically, we define the prediction to be true if $IoU > 0.5$; otherwise, we take it as false (Nowozin, 2014).

For the other layers in this model, 'predict' is used for estimating the bounding box, 'pre-valid' is used for calculating the accuracy of pre-trained and the 'valid' is for calculating the accuracy of formal training.

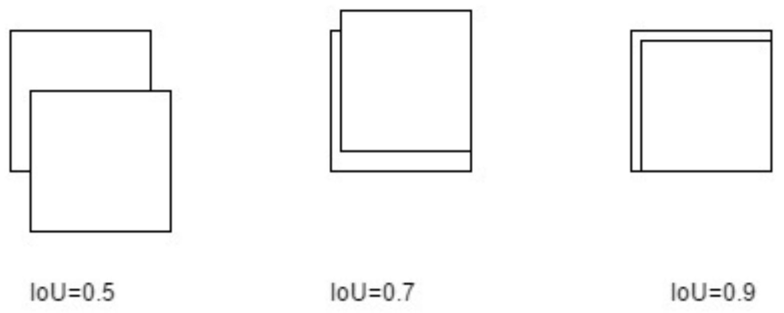


Figure 3.6.4 Using IoU to define the accuracy of bounding box predicted

3.7 Training Strategy

For training phase, we have two different training methods: pre-training and formal training. YOLO segmented an image into a 7×7 region. The pre-training is a kind of classification problem, which is used to find out which grid is the center of the detected object.

For training dataset, we prepare for 1720 images of fire flames. We set every 172 images as one epoch and trained 20 epochs for pre-training phase and 40 epochs for formal training phase.

The first nine convolutional networks output one $7 \times 7 \times 1024$ feature matrix. Through one more convolution in "conv_pre", it will become one 7×7 matrix which could be thought as a 49D vector. The possibilities of positioning in the center grid will be set as $y = (y_1, y_2, y_3 \dots . y_{49})$. Normalize them by using Softmax, we get $p = (p_1, p_2, p_3 \dots . p_{49})$. The p_i is presented as:

$$p_i = \frac{\exp(y_i)}{\sum_{k=1}^{49} \exp(y_k)} \tag{3.13}$$

If the ground truth is in the jth grid, then we will get $q_j = 1$ from $q = (q_1, q_2, q_3 \dots . q_{49})$. If $i \neq j$, $q_j = 0$. The loss function of pre-training will be:

$$L_{pre} = \sum_{k=1}^{49} -q_k \log p_k \tag{3.14}$$

The pre-training aims at minimizing loss function L_{pre} and increasing the accuracy of prediction.

The second phase is formal training. Formal training is to ensure the width and height of the bounding boxes. Let us assume the width and height are (w, h), and $(\hat{w}, \hat{h})$ for the ground truth. The loss function is:

$$L = (w - \hat{w})^2 + (h - \hat{h})^2 + L_{pre} \tag{3.15}$$

The loss function is based on the sample definition. In the real training phase, we used min-batch, which transmits *N* (batch size) samples to be trained. Hence, the real loss function will be the average of all the sample functions,

$$\tilde{L}_{pre} = \frac{1}{N}\sum_{j=1}^{N}\sum_{k=1}^{49} -q_{jk}\log p_{jk} \tag{3.16}$$

$$\check{L} = \frac{1}{N}\sum_{j=1}^{N}\left[(w_j - \hat{w}_j)^2 + (h_j - \hat{h}_j)^2\right] + \tilde{L}_{pre} \tag{3.17}$$

3.8 Training Dataset

For the training data, we have 172 images of fire flames, we used image processing functions such as flipping, adjusting brightness and saturation to create 10 samples from each image. These samples are different because they have differences at the pixel level and also the positions of bounding boxes are also different. It could reduce time cost of collecting training data. So, we get 1720 training images.

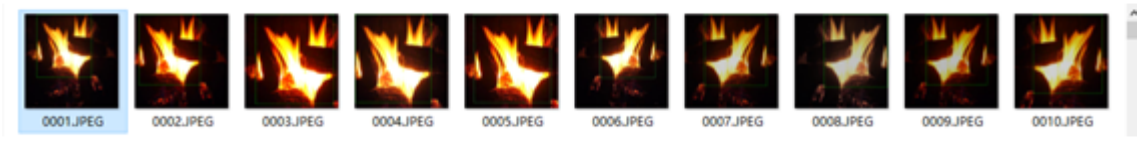


Figure 3.8.1 Training samples based on image enhancement

We now have 20 epochs for pre-training and 40 epochs for formal training, which means the total training set has 60 epochs, 10320 training times.

Chapter 4
Results

The main content of this chapter is to illustrate the result of all methodology used for flame detection in this report, which is the color based model, a combination of color and HSI based model, and CNN. The result will be demonstrated in this chapter. Also, for deep learning method, the training phase would be introduced at last to evaluate the method we used.

As the color model and CNN method introduced, the result is demonstrated in this chapter. There are four methods to detect fire flames from digital images.

4.1 Color Model Result

There are two different color based models that have been adapted in this report. We use the combination of RGB and HSI based model to overcome the disadvantages of the single-color based model. The original image is shown as,

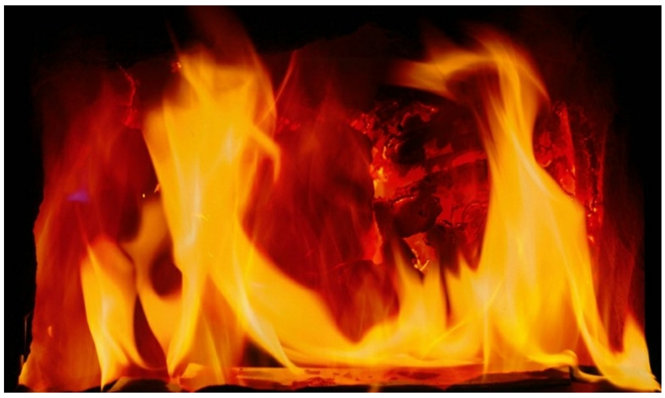


Figure 4.1.1 The image for testing three different methods

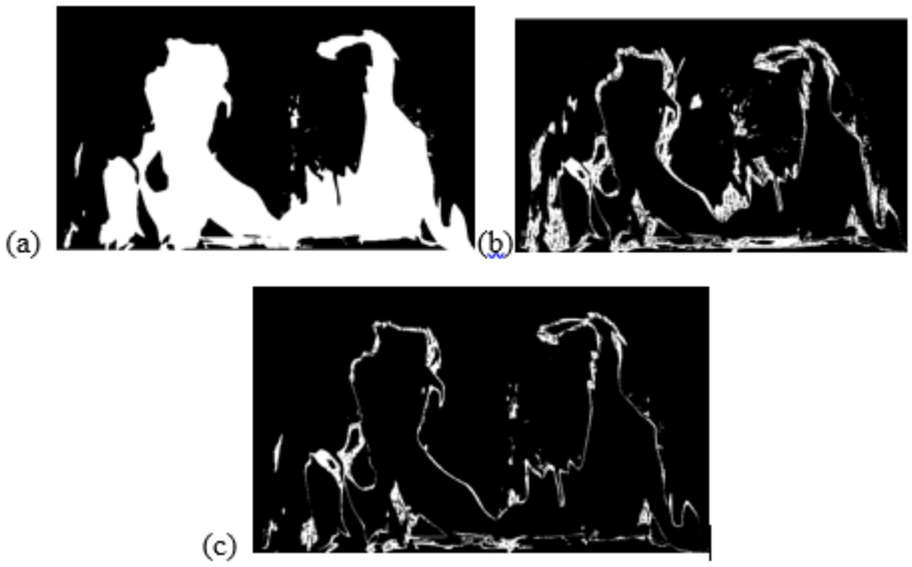


Figure 4.1.2 Results of fire flame detections

Fig.4.1.2 (a) shows the fire region by using the RGB color based model, we got the full flame region. The result perfectly catches the fire flames from those digital images. Figure

4.1.2 (b) is the result from HSI color based model, from Figure 4.1.2(b), we get the shape of the fire. If we want to get the region of fire flames using HSI based model, the thresholds for each channel need to be adjusted.

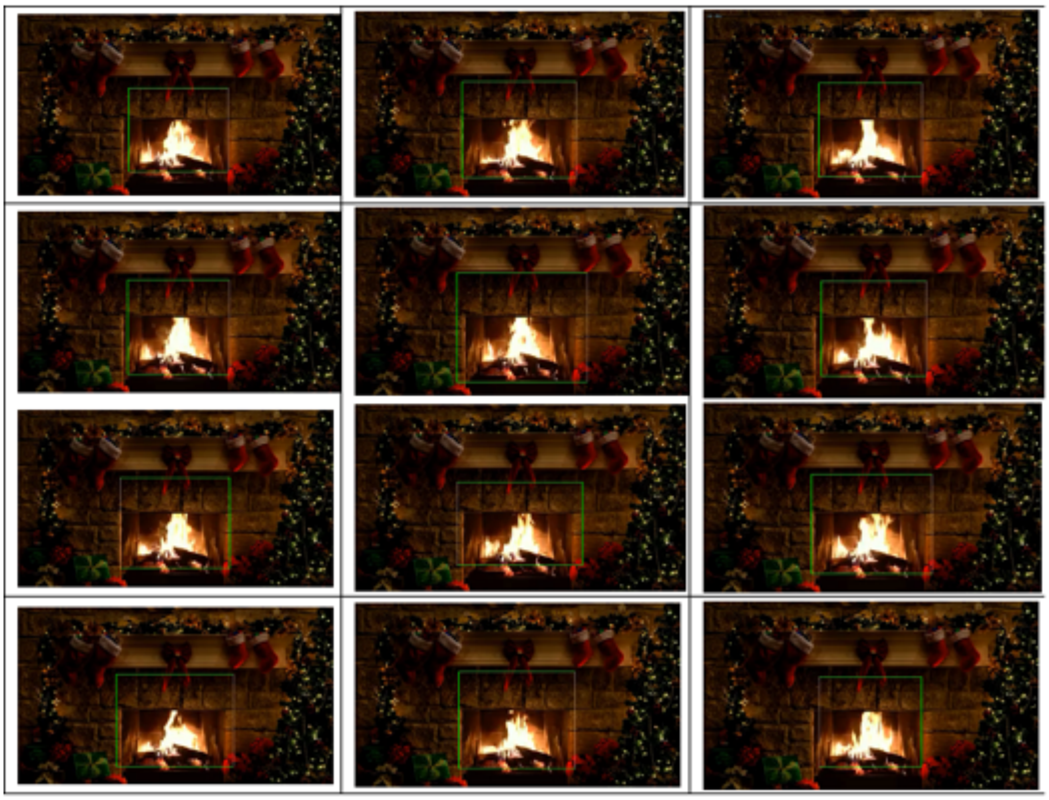
Figure 4.1.2 (c) is a combination of both RGB and HSI based model which reflects the shape of the fire flames but with better performance comparing to HSI based model.

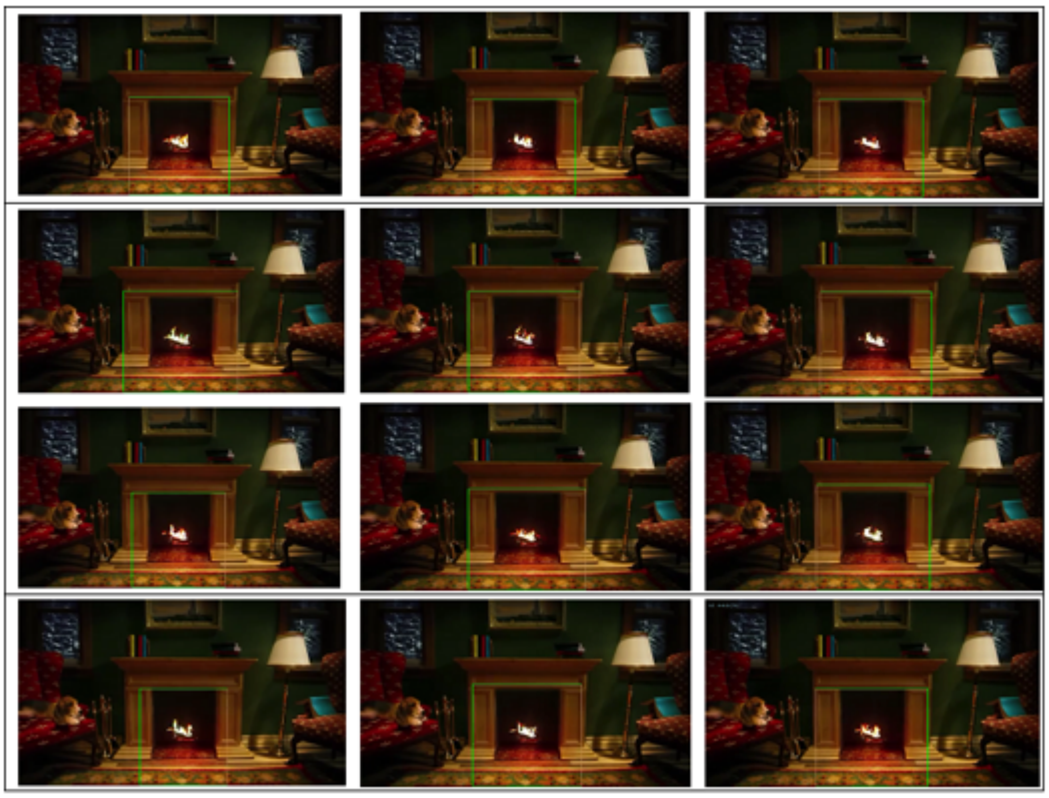
4.2 Detection Results from CNN

In the flame detection method based on convolutional neural network, the training set is collected from the Internet which includes 172 pictures of fire flames. As stated in Chapter 3, we used the techniques of image enhancement, which includes picture rotating,flipping, and bright adjusting so as to increase a number of images of our training set.

a.

b.

c. 

d. 

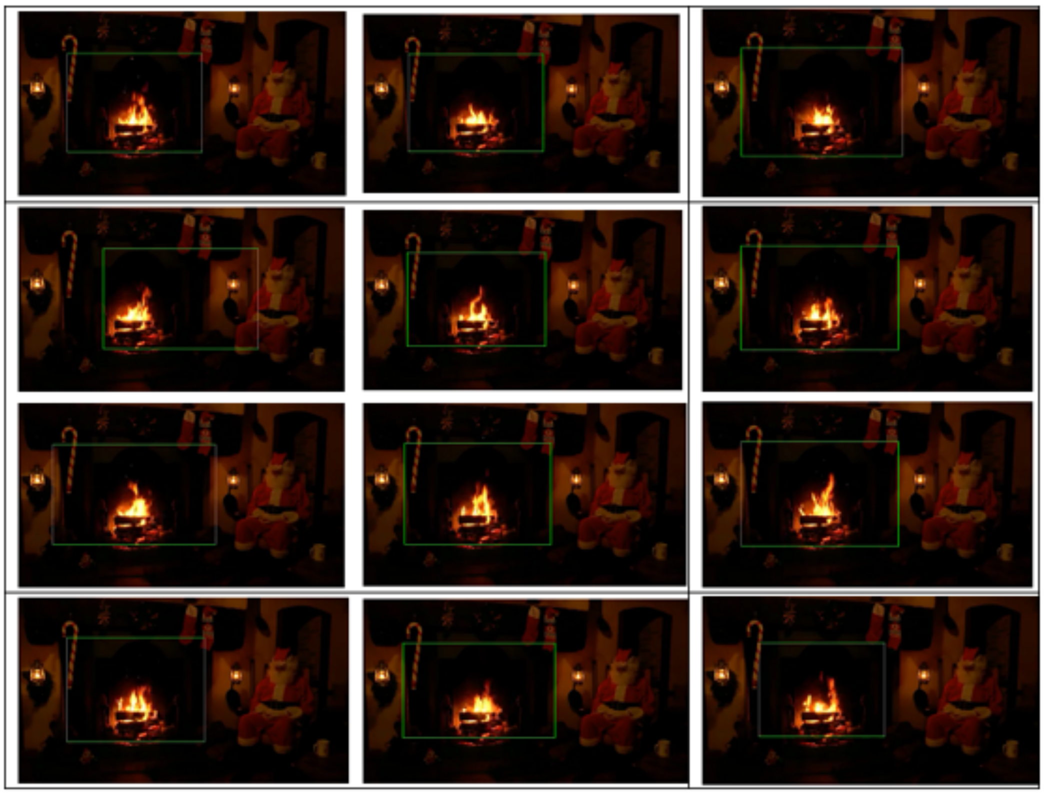
e. 

Figure 4.2.1 Motion pictures of fire detection using CNN

The convolutional neural network has a completely different way to extract features comparing to the color based model. Therefore, the performances still seem same. The flame center has been detected although some of the center shift a little bit of the real center position. From the result of these motion pictures, it demonstrates that in most of the situations, the detection areas are very precise, but when there are lots of interference elements, the bounding box could be shift from the correct position.

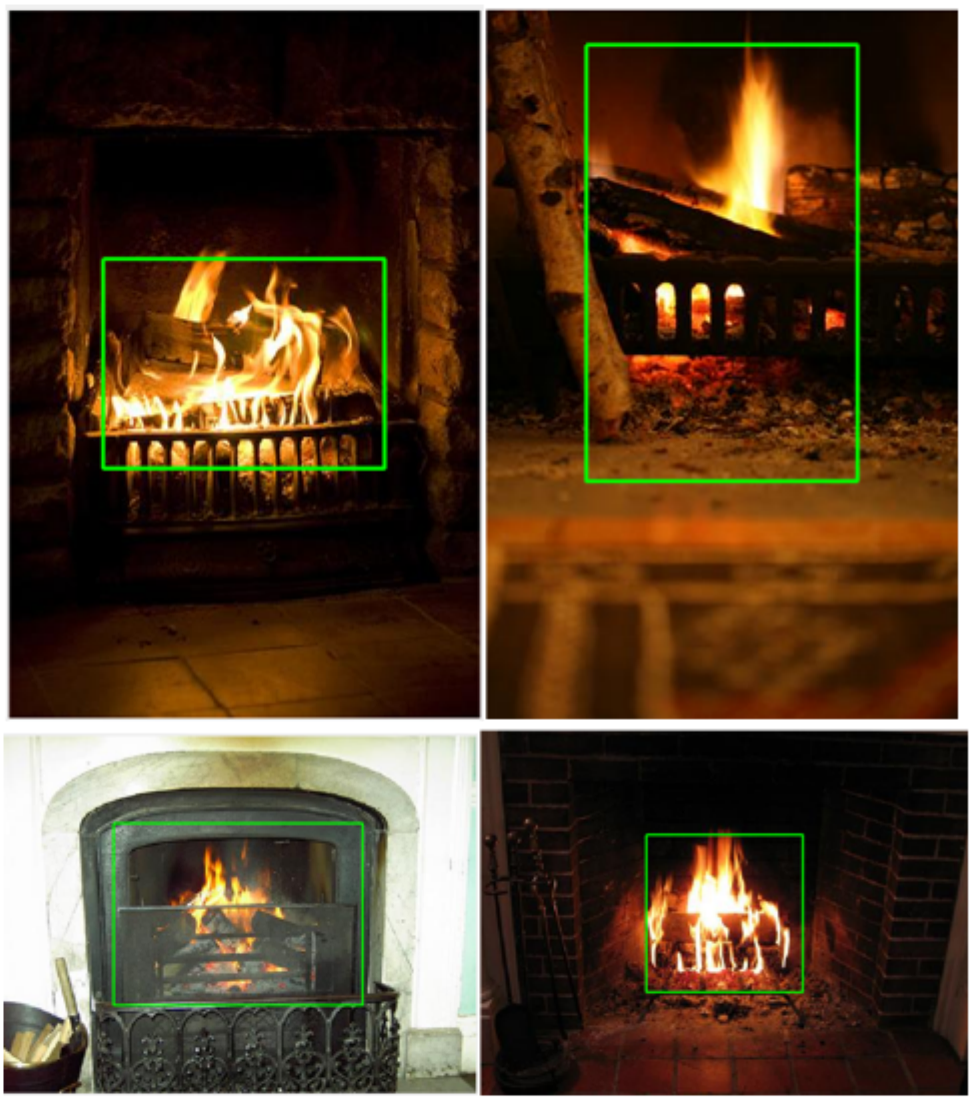


Figure 4.2.2 The detected flame center

Figure 4.2.2 shows some of the results of CNN model in images. When the images which do not have too many interferences, we can see the box exactly displays the whole fire area.

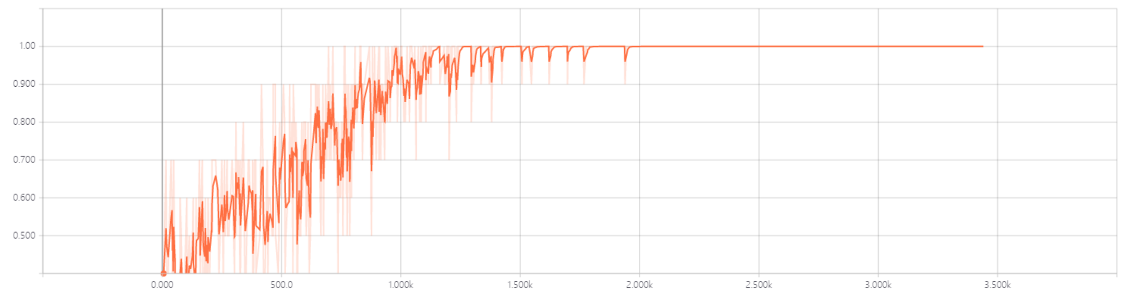


Figure 4.2.3 Pre-training accuracy

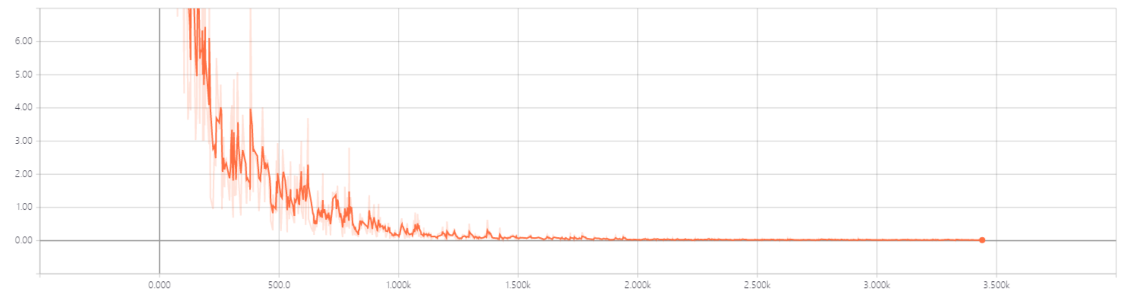


Figure 4.2.4 Pre-training loss

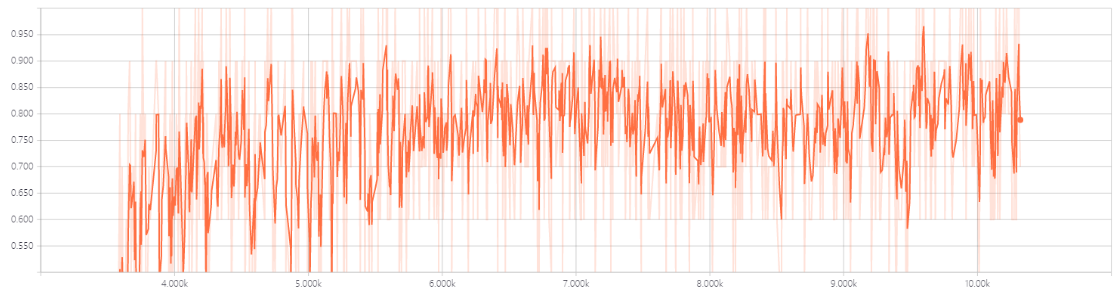


Figure 4.2.5 Formal-training accuracy

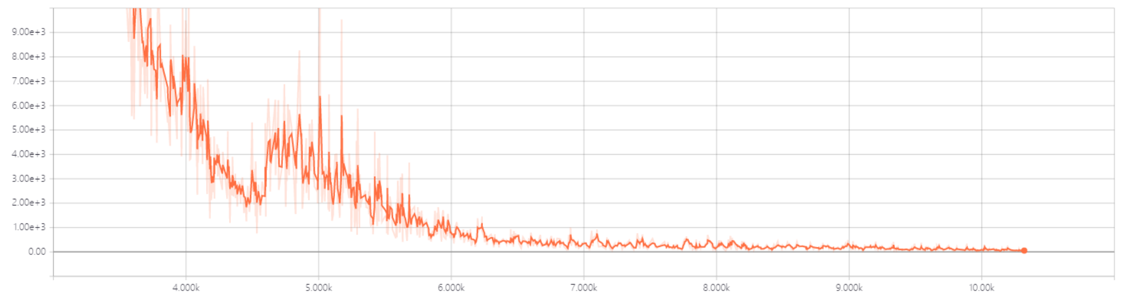


Figure 4.2.6 Formal-training loss

Figures 4.2.3 ~ 4.26 shows the loss and accuracy at pre-training and formal phases, respectively. After more than 2000 times training, the accuracy of pre-training is extremely close to 100%, and the accuracy of formal training is still not stable after 6000 times training. The final accuracy is close to 81%. For YOLO, it is actually not a really bad result.

Chapter 5
Analysis and Discussions

In this chapter, the discussion and resultant analysis with respect to outcomes of the experiments are clearly demonstrated and presented. The results of shallow learning and deep learning will be compared in this chapter, and the evaluation of these two methods will be discussed. The limitations of these methods will also be stated finally.

5.1 Evaluations of Deep Learning Method

In deep learning, the training time really depends on the performance of the devices. When using CPU to training the dataset, it could take approximately 50 times than using GPU.

Table 5.11.1 Comparing time cost by using CPU to GPU

Device	Samples	Epoch	Time cost
CPU	1720	2	12 hours
GPU (NVIDA GTX 970M)	1720	6	90 minutes

In Chapter 4, we mentioned that the increase of interference may decrease the accuracy of the box position. We can see a and b in figure 4.2.1, the interference elements in these images would be the light bulbs ahead of the fireplace. Even the situation between a and b are pretty similar, but the result of a are better than b’s. The image of b has more interference elements than a, and the left side of the light are being circled which decrease the accuracy of the detection.

Meanwhile, when the background is a little bit easier, the accuracy would perform much better (c and d). In the image of e in Figure 4.2.1, there is one light at the right side of flame center. In the demo video, the light has a very little chance to create false alarms. The video b has the lots of false alarms in all the demo. For evaluating all demo video, over 80% of bounding boxes get the right positions of the flame center.

When comparing to the shallow learning method which used the color model. The deep learning method is more robust when using in some complicate situations. When the flame is not in the common situation such as gas fire, which means it could have completely different color features. When these kinds of fire flames used in the color model, the performance cannot reach the deep learning method’s. With setting up the proper training set, the performance of deep learning method can ignore the different kinds of flame influence which may increase when used in the complicate situation.

When comparing to the other object detection methods using deep learning, such as using YOLO or Fast-RCNN. When comparing to other object detect accuracy, The YOLO has one 81.4% accuracy on cat, 68.3% on bus, 77.2% on dog, 63.5% on person. The model we used in this

5.2 Limitations

The limitation of the traditional fire detection method is that it requires the developers have the specific knowledge of fire flames, such as motions, color information and patterns of fire flames. With the knowledge to set up one better model helps to increase the performance of model to get a higher performance on accuracy.

However, most of the time we do not have that knowledge, this may influence a lot when we create the model. Even if we have the specific knowledge of this domain, when the object changes, the models need to be rebuilt for adapting new objects.

Deep learning requires the knowledge of machine learning and neural networks. It does not require additional knowledge if the training set possesses the target object. The model does not need to change to detect new objects. Also, it could be used for multiple object detection. But the current deep learning for object detection still cannot reach a very high accuracy.

In this report, one optimized YOLO model is used. YOLO has an excellent performance on decreasing training time. Meanwhile, because of setting up the 7×7 grid cells, the bounding boxes do not catch the exact region of fire flames.

The biggest advantage of YOLO is the training speed (with setting less proposals at first), whether the other framework such as Fast-RCNN, the accuracy using that framework could perform much better.

Another limitation of deep learning in this report is the dataset. There is only one object needed to be detected, but the training set is still too small. Even with the enhanced image data set, there are only 1720 images of fire flames, compared to other deep learning training set, it is still too small. Also, the training set are mostly based on flame in fireplace. If there are more other flames such as candle fire, stick fire, forest fire and building fire, the accuracy may increase in the other situations. Even fire identification could be implemented.

Chapter 6
Conclusion and Future Work

In this report, through analysis and evaluation of fire detection by using shallow learning and deep learning methods, the last chapter is going to present the conclusion. With the limitations of these methods, the possible future work will be presented at the end of this report.

6.1 Conclusion

There are two main methods which have been used in this report to identify fire flames from digital images: shallow learning (color based model) and deep learning model. Both methods performed well on flame detection, whether the deep learning has a better performance than that of the color based model, it still needs to be tested using a larger test set.

There are two main features of fire flames, one is color and the other is temperature. Deep learning can use them to find and extract graphical features from the images. At present, deep learning still cannot recognize whether a flame is a real fire flame or a fake fire flame created by computer graphics.

The different kinds of flames have different color model. When using one color model using at some other flames, the accuracy may decrease a lot. As mentioned in Chapter 1, the most region of fire flame are based on the figure 1.1.1, but there are many other flames such as gas fire (Figure 6.1.1). The disadvantage of the color model is that the model is not suitable for all kinds of flames. When the test set are more complicate, such as lots kinds of fire flame, the performance of color model will not perform as good as before.

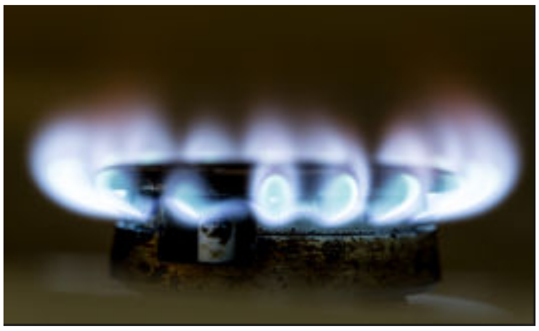


Figure 6.1.1 The gas fire

While using the deep learning method, with setting the training set using different kinds of flame, the performance of detection will not be affected by different kinds of color. But using the color model, such as the model we used in Chapter 3, the gas fire could not be detected by using that color model.

On the other hand, the method of flame detection for deep learning in this report only used CNN. There are a few of deep learning methods for digital image processing, such as auto encoder, sparse coding, restricted Boltzmann machine, deep belief network and recurrent neural network. We will combine them together to develop a better approach for fire flame detection (Ba, Mnih & Kavukcuoglu, 2014).

6.2 Future Work

Our future work includes,

(1) We will work for improving the CNN model to increase the accuracy of fire detection. Other CNN models (Fast RCNN, Faster RCNN) could have a better performance on accuracy even if they may take more time on training. How would the performance of fire detection be if we sacrifice the training time in this area?

(2) The combination of YOLO and Fast RCNN model together (Redmon et al. 2016) will have a better performance than YOLO but it will cost more training time. The other evolution model of YOLO (Redmon & Farhadi, 2016), YOLOv2, could reach 78.6 mAP on PASCAL VOC 2007.

(3) Not only the flame detection but also the other object detection by using deep learning model would be better in performance when the deep learning models are improved. Even if deep learning may not have a higher accuracy than shallow learning, with the development of deep learning techniques, we are confident that this situation will be completely improved in future.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016). TensorFlow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA.

Ba, J., Mnih, V., & Kavukcuoglu, K. (2014). Multiple object recognition with visual attention.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., & Bengio, Y. (2010). Theano: A CPU and GPU math compiler in Python. In Proc. 9th Python in Science Conf (pp. 1-7).

Celik, T. (2010). Fast and efficient method for fire detection using image processing. ETRI journal, 32(6), 881-890.

Celik, T., & Demirel, H. (2009). Fire detection in video sequences using a generic color model. Fire Safety Journal, 44(2), 147-158.neural networks. In Advances in neural information processing systems (pp. 1097-1105).

Celik, T., Demirel, H., Ozkaramanli, H., & Uyguroglu, M. (2007). Fire detection using statistical color model in video sequences. Journal of Visual Communication and Image Representation, 18(2), 176-185.

Celik, T., Ozkaramanlı, H., & Demirel, H. (2007). Fire and smoke detection without sensors: image processing based approach. In European Signal Processing Conference (pp. 1794-1798). IEEE.

Chan, T. H., Jia, K., Gao, S., Lu, J., Zeng, Z., & Ma, Y. (2015). Pcanet: A simple deep learning baseline for image classification?. IEEE Transactions on Image Processing, 24(12), 5017-5032.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., & Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems.

Chen, T. H., Wu, P. H., & Chiou, Y. C. (2004). An early fire-detection method based on image processing. In International Conference on Image Processing (Vol. 3, pp. 1707-1710). IEEE.

Chunyu, Y., Jun, F., Jinjun, W., & Yongming, Z. (2010). Video fire smoke detection using motion and color features. Fire technology, 46(3), 651-663.

Ciresan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2011). Convolutional neural network committees for handwritten character classification. In International Conference on Document Analysis and Recognition (ICDAR) (pp. 1135-1139). IEEE.

Collobert, R., Bengio, S., & Mariethoz, J. (2002). Torch: a modular machine learning software library (No. EPFL-REPORT-82802). Idiap.

Dahl, G. E., Sainath, T. N., & Hinton, G. E. (2013, May). Improving deep neural networks for LVCSR using rectified linear units and dropout. In IEEE International Conference on Acoustics, Speech and Signal Processing (pp. 8609-8613). IEEE.

Gidaris, S., & Komodakis, N. (2015). Object detection via a multi-region and semantic segmentation-aware cnn model. In IEEE International Conference on Computer Vision (pp. 1134-1142).

Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1440-1448).

Giusti, A., Ciresan, D. C., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2013, September). Fast image scanning with deep max-pooling convolutional neural networks. In IEEE International Conference on Image Processing (pp. 4034-4038). IEEE.

Graham, B. (2014). Fractional max-pooling.

Hanamaraddi, P. M. (2016). A Literature Study on Image Processing for Forest Fire Detection. IJITR, 4(1), 2695-2700.

He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In European Conference on Computer Vision (pp. 346-361). Springer International Publishing.

Hecht-Nielsen, R. (1988). Theory of the backpropagation neural network. Neural Networks, 1(Supplement-1), 445-448.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., & Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In ACM international conference on Multimedia (pp. 675-678). ACM.

Jun, C., Yang, D., & Dong, W. (2009, March). An early fire image detection and detection algorithm based on DFBIR model. In World Congress on Computer Science and Information Engineering (Vol. 3, pp. 229-232). IEEE.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188.

Kawa, H., Khartade, A., Sonawane, S., & Madole, S. (2016). Smart Fire Detection System using Image Processing. International Journal of Engineering Science, 5485.

Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

Ko, B. C., Cheong, K. H., & Nam, J. Y. (2009). Fire detection based on vision sensor and support vector machines. Fire Safety Journal, 44(3), 322-329.

Kong, S. G., Jin, D., Li, S., & Kim, H. (2016). Fast fire flame detection in surveillance video using logistic regression and temporal smoothing. Fire Safety Journal, 79, 37-43.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

Kumar, A., Sujith, P., & Veeramuthu, A. (2016). Optical Flow Reckoning for Flame Disclosure in Dynamic Event Using Hybrid Technique. In International Conference on Soft Computing Systems (pp. 861-871). Springer India

Lawrence, S., Giles, C. L., Tsoi, A. C., & Back, A. D. (1997). Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks, 8(1), 98-113.

LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10), 1995.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

Liu, C. B., & Ahuja, N. (2004). Vision based fire detection. In International Conference on Pattern Recognition (Vol. 4, pp. 134-137). IEEE.

Marbach, G., Loepfe, M., & Brupbacher, T. (2006). An image processing technique for fire detection in video images. Fire safety journal, 41(4), 285-289..

Mathi, P. T., & Latha, L. (2016) Video Based Forest Fire Detection using Spatio-Temporal Flame Modeling and Dynamic Texture Analysis.

Matsugu, M., Mori, K., Mitari, Y., & Kaneda, Y. (2003). Subject independent facial expression recognition with robust face detection using a convolutional neural network. Neural Networks, 16(5), 555-559.

Mozer, M. C. (1989). A focused back-propagation algorithm for temporal pattern recognition. Complex systems, 3(4), 349-381.

Nowozin, S. (2014). Optimal decisions from probabilistic models: the intersection-over-union case. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 548-555).

Qi, X., & Ebert, J. (2009). A computer vision based method for fire detection in color videos. International journal of imaging, 2(S09), 22-34.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 779-788).

Redmon, J., & Farhadi, A. (2016). YOLO9000: Better, Faster, Stronger. arXiv preprint arXiv:1612.08242.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural networks, 61, 85-117.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Smith, L. N., & Topin, N. (2016). Deep convolutional neural network design patterns. arXiv preprint arXiv:1611.00847.

Sun, Y., Wang, X., & Tang, X. (2014). Deep learning face representation from predicting 10,000 classes. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 1891-1898).

Toreyin, B. U., Dedeoglu, Y., & Cetin, A. E. (2005). Flame detection in video using hidden markov models. In IEEE International Conference on Image Processing (Vol. 2, pp. II-1230). IEEE.

Toreyin, B. U., Dedeoglu, Y., Gudukbay, U., & Cetin, A. E. (2006). Computer vision based method for real-time fire and flame detection. Pattern recognition letters, 27(1), 49-58.

Toulouse, T., Rossi, L., Celik, T., & Akhloufi, M. (2016). Automatic fire pixel detection using image processing: a comparative analysis of rule-based and machine learning-based methods. Signal, Image and Video Processing, 10(4), 647-654.

Vicente, J., & Guillemant, P. (2002). An image processing technique for automatically detecting forest fire. International Journal of Thermal Sciences, 41(12), 1113-1120.

Vijaylaxmi, V. K., & Sajjan, S. C. (2016). Fire Detection using YCbCr Color Model.

Wan, J., Wang, D., Hoi, S. C. H., Wu, P., Zhu, J., Zhang, Y., & Li, J. (2014). Deep learning for content-based image retrieval: A comprehensive study. In ACM international conference on Multimedia (pp. 157-166). ACM.

Wang, L., Li, A., Yao, X., & Zou, K. (2016). Fire Detection in Video Using Fuzzy Pattern Recognition. In International Conference on Oriental Thinking and Fuzzy Logic (pp. 117-127). Springer International Publishing.

Yang, Y., Yuhua, P., & Zhaoguang, L. (2007). A fast algorithm for YCbCr to RGB conversion. IEEE Transactions on Consumer Electronics, 53(4).

Zeng, D., Liu, K., Lai, S., Zhou, G., & Zhao, J. (2014, August). Relation Classification via Convolutional Deep Neural Network. In COLING (pp. 2335-2344).