Fire detection based on convolutional neural networks

Sungjae Park
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
park1637@morris.umn.edu

ABSTRACT

Early fire detection and generating fire alarm on time is a crucial need to prevent life and property damage. Many studies have been focused on fire detection with surveillance using convolutional neural network as the majority of the activation of fire alarm systems are established on the basis of camera. However, this method has limitations as it requires a lot of computational time and memory. In this paper, we will primarily focus on the two recent cost-effective and more accurate fire detection CNN in surveillance system.

Keywords

Machine Learning, Fire Detection, CNN, GoogleNET

1. INTRODUCTION

Surveillance devices are widely used in a variety of areas such as e-health, monitoring, and autonomous driving due to increase in processing capabilities of smart devices [5]. Surveillance devices have many applications in our lives due to their abilities to detect various accidents and incidents such as fire, natural disaster, crash, and medical emergency which require a quick response to reduce further disasters. Fire is one of common risks, and early detection can reduce casualties and risk of property damage. In this paper, we will mainly focus on the fire detection in surveillance videos and introduce two different approaches of fire detection based on convolutional neural networks by Dung and Ro [3] and Muhammad et al [5]. Background information regarding how convolutional neural network work is given in section 2, and section 3 introduces two proposed architectures of CNN to detect fire with high accuracy. Experimental results from the two approaches are given in section 4, and conclusion of this paper and further approaches regarding fire detection model are given in section 5 [2].

2. BACKGROUND

Manual fire detection from video is monotonous and nuisance work since it is time-consuming and ineffective. Fire detection using sensors in general has disadvantages because it does not provide information about the location of the initial fire, the direction of smoke propagation, or the magnitude of the fire. Many studies have been focused on fire detection with surveillance using convolutional neural network because surveillance devices with image and video can remedy the disadvantage of sensors detection method. Through the construction of fire detection engineering, it is easier to capture an outbreak of fire. The most well-known algorithm in application of fire detection system is Convolutional Neural Network (CNN). It is particularly used for flame detection at early stages from surveillance videos or images. In operating the fire detection tests, the layers are considered as important keys of simulation results and there are three main layers existed: Convolutional layer, Stride, and Pooling layers which are valuable to recognize the features of adjacent images with maintaining the image spaces and possible to learn or extract the features of images through the multiple filters. Colors and flame flickering are significant factors in terms of fire detection. It is required to distinguish the fire and non-fire situations for more accurate fire detection based on the image pixels and frequency of flickering. Although numerous image processing methods of fire detection are proposed with achievement of considerable success, there are still drawbacks of their performances. In this paper, we compare and analyze the performance of classifying models specifically optimized for pixel based classification.

[7][4]

2.1 Artificial Neural Network (ANN) and Deep Neural Network (DNN)

Artificial Neural Network(ANN) is a computational model which is inspired by the biological neural networks and is intended to replicate the methods that human learns [8]. It consists of input, output, and hidden layers as shown in the figure 1. The input layers provide information from the external environment to the neural networks and there is no computation is executed in any of the input layers. The main role of hidden layers is computing and transforming the input information in order to achieve the desired outputs. It is an appropriate tool for finding patterns or organizing images into correct outputs. The output layers are responsible for collecting and transferring information that it has been designed to provide. While the ANN process appears to be flawless, misidentifications could happen and lead to produce false outputs. Therefore, A technique called backpropagation is developed and becomes a major part of artificial intelligence which allows networks to adjust the hidden layers where the outputs are not consistent with expected outcome [8]. Deep Neural Network(DNN) is similar to ANN such as have multiple layers and simulating the human brain activities in particular collecting, transmit-

UMM CSci Senior Seminar Conference, April 2018 Morris, MN.

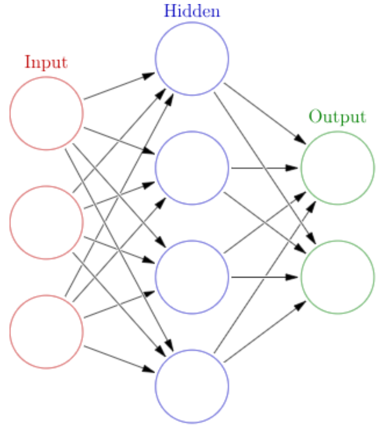

Figure 1: Artificial neural network[8]

ting, and recognizing inputs through various layers [10]. It uses term deep, because there are more than 2 hidden layers and can process more complex tasks. It consists of artificial processing units that are arranged in hidden layers. DNNs learn object detections through training with massive dataset, during training, the DNN acquires features from the data to perform the given tasks without manual works of feature extraction [10]. Convolutional neural network (CNN) is one of the famous method in deep neural networks. Information on CNN is discussed in the next paragraphs, including the definition of CNN, layers, and how they works in terms of image recognition or classification.

2.2 Convolutional Neural Network(CNN)

CNN is one of Artificial Neural Network (ANN) and has become one of the most prominent Deep Neural Network in various fields through many years of research, such as object, pattern, action, and text recognition, visual saliency detection, and more [2]. Classical ANNs were not able to solve complex tasks in large models. However, CNN could solve such problems by having spatially independent features. For example, it is possible to detect faces without knowing the location of the faces. CNN has another important feature in image classification which is to acquire abstract features when input moves to the deeper layers. In the example of image classification in the figure 2, the first layer will be the edges of the image, the second layer is more simple aspects rather than first one, and higher level traits of face are in the next layers. CNN has many layers: Convolutional layer, pooling layer, and fully connected layer [1]. They are explained in the next section.

2.2.1 Convolutional Layer

Convolutional layer is a first hidden layer in CNN and uses a convolution operation to the input image and connects to the neurons in the next layer. So imagine there is an input image of a dataset with a width and height of 32x32 pixels, and a depth of 3 that are RGB channel. RGB channels are red, green, and blue channels that constitute images. The input then consists of 32x3x3 which is two dimensions and 3 RGB channels of neural network. If there is another neuron added into the hidden layer, we need another 32x32x3 weight connection, therefore, total becomes 32x32x3x2 which is more than 6000 weight parameters used

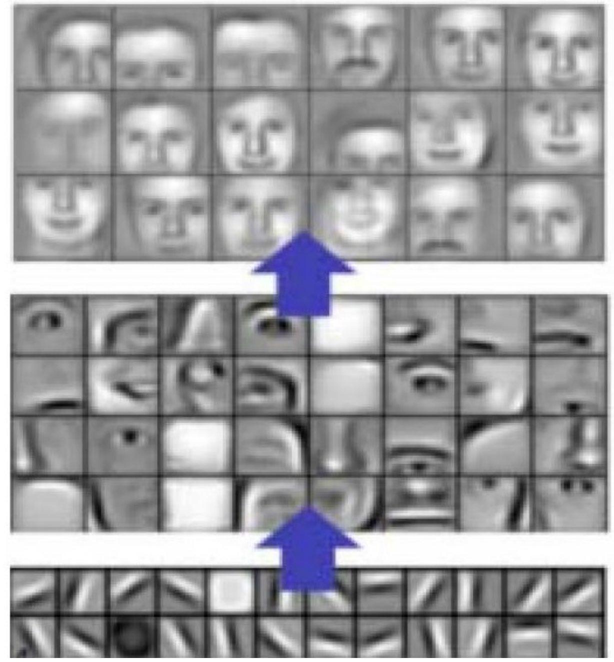

Figure 2: feature learning in each layer [1]

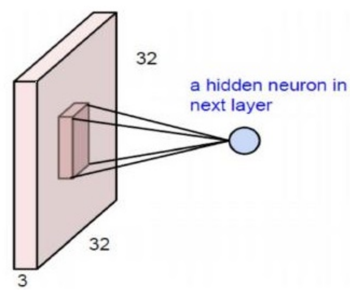

Figure 3: Connection of local region to the next layer[1]

to connect to two neurons. It is important to have the same values of width and height in the next layers. In this case, the networks need 32x32x3 by 32x32 which is 3,145,728 connections [1]. Thus, we look for more efficient way, find local regions instead of a whole image. Figure 3 explains the connection of local regions and the next layers. This means the hidden neuron in next layer works only with the same region of the previous layer. For instance, only 5x5 neurons can be connected, so adding 32x32 neurons in the next layer need 5x5x3 by 32x32 connections which is 76,800 weight connections [1]. It is noticeable that the size of connections decreased in local connection, but there is an assumption for even more reducing parameters. Keep fixed weights of local connection, apply the whole neurons of the next layer, and then only 75 weights need to connect the next layer. Figure 4 indicates many layers with their own filters that process the same region of the input image [1].

2.2.2 Stride and Zero-Padding

After deciding the filter size, we choose stride and padding

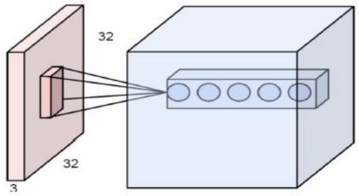

Figure 4: Multiple layers with different filters but looking at the same region[1]

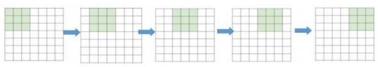

Figure 5: Explain of Stride 1[1]

for controlling the volume of output by moving the filter. For example, figure 5 illustrates stride 1 that the filter moves one unit in each time and applies convolution matrix to each portion of the region. There is a 7x7 input volume with a 3x3 filter, and if we move the filter one unit every time, the output consists of 5x5 volume in the end. If we move the filter and make stride 2, the output volume is 3x3 since the filter shifts 2 units at a time. From these examples, increasing the stride gives diminished output volume. However, we will face space issues if we try stride 3. The reason is that the receptive fields no longer fit on the input volume. In order to solve the problem of losing the information which exists in the border that the filter does not slide and shrinking the output size with depth, zero-padding is an efficient method. Figure 6 indicates zero-padding which adds zeros around the input borders to keep the input volume as original, so that we keep the information as much as we can and extract the low level features in the early layers [1].

2.2.3 *Pooling Layer*

The purpose of pooling layer is to reduce number of dimensions of data and maintain the important information. It is also called downsampling. When it comes to image processing, the pooling layer can be used to reduce the image resolution. Max pooling is one of the most widely used pooling layer method. In figure 7, it shows the max pooling with 2x2 filter and applies stride 2 for downsampling from 2x2 blocks to 1 block. More specific, there are sub regions with different colors in rectangles (pink, green, yellow, blue), and 2x2 filter is used since it is the most common size in max pooling. It takes the largest value of each rectangles in sub-region. Stride 1 also can be used for downsampling prevention, but it is important to consider that downsampling does not maintain the spatial information. [1].

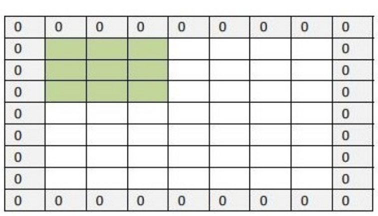

Figure 6: zeropad [1]

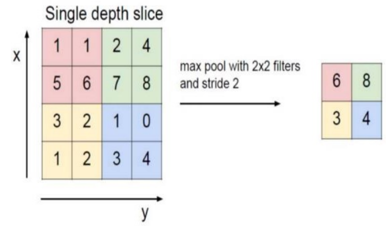

Figure 7: max-pooling [1]

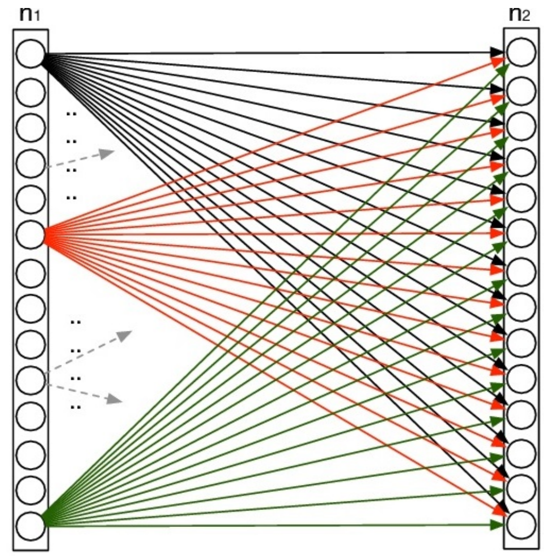

Figure 8: Fully-Connected Layer[9]

2.2.4 *Fully Connected Layer*

Fully connected networks are neural networks such that each neuron is connected to every neuron in the previous layer, and connection has its own weight as shown in Figure 8 [1]. This layer combine the number of convolution and pooling layers so it produces a high level representation of the input data. The purpose of the fully connected layer is to use features of the output from the convolutional and pooling layers for classifying the input image into various classes based on the training dataset. It is generally used in the last hidden layer of the CNN to connect to the output layer and construct the desired number of outputs. Since each of the nodes in the last frames are connected as a vector to the first one, it takes a long time in processing the CNN [1].

3. METHODS

Dung and Ro [3] carried out a fire detection algorithm using a surveillance camera system. Fire flames in candidate areas are identified by temporal analysis with image pixels of the colors and flickering of fire frames. Also, they developed a fire classifier using a cascade model as shown as figure 9 that enable to detect true fire and non-fire areas rapidly and accurately. The architecture of the algorithms consists of three phases: fire areas detection, identifying fire and non-fire areas, and temporal analysis for ultimate recognition of a fire alarm need [3].

3.1 Detection of candidate fire areas

To detect candidate fire regions, Dung and Ro [3] used RGB color map which constructs all the colors from the combination of the Red, Green and Blue and the flickering energy map which is a method of perceiving fire by the frequency of luminance flickering to estimate every image pixel as they are important factors of recognizing possible fire regions. Possible fire colors in RGB model are described by the comparison of the color of each pixel, and an image pixel is classified as a potential fire if it satisfies the following conditions [3]:

$$R \geq G \geq B$$

$$R > R_T$$

$$S > (255 - R)S_T / R_T$$

As shown above in the formulas, R, G, and B stand for red, green and blue color of image pixels. In the fire classification, Red channel of fire pixels should be higher than their green channel and the green channel should be higher than the blue channel. S signifies a saturation and it is an expression for the relative bandwidth of the visible output from a light source as the fire. RT , ST are respectively two thresholds for R and S color channels [3]. In addition, not only comparison of the color of each pixel is satisfied to detect fire, but also fire flickering is an important key of recognize fire or flames. The fire flickering detection method estimates the cumulative frequency of fire flickering by the following equations [3]:

$$dI(t) = I(t) - I(t-1)$$

$$E(t) = \alpha E(t-1) + (1-\alpha) dI(t)$$

$dI(t)$ is detecting intensity which is calculated by the intensity of the image pixels at time t subtracts the intensity of previous time and E(t) represent cumulative flicker energy [3]. Following these methods, it is possible to estimate fire pixel through color and cumulative flicker energies whether the input data is a potential fire or definite non-fire objects [3].

3.2 Fire classification

The pixel-level methods are effective to flame detection, but they are easy to provide false decision since there are many movements that would have possibly recognized as false flickering as similar features [3]. Thus, Dung and Ro [3] introduced the cascade model as a practical model for combining multiple features into one classifier. To calculate the moving distance of the object for the period of time, the equation below is used in [3]:

$$D(t) = \sqrt{(x_t - x_0)(x_t - x_0) + (y_t - y_0)(y_t - y_0)}$$

Where x_0 and y_0 are the first position of each x-axis and y-axis of the object and xt and yt for the current position of x-axis and y-axis respectively after a period of time [3]. It is important to point out that a larger threshold should be selected to eradicate only low probability of fire regions [3].

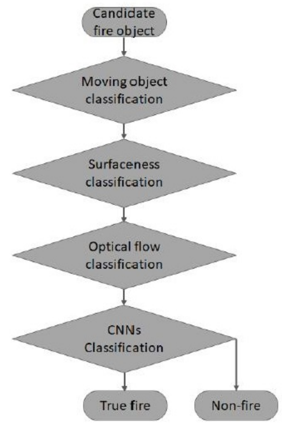

Figure 9: The flow chart of fire classification cascade model[3]

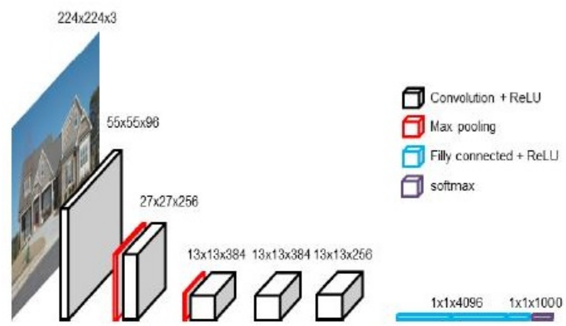

Figure 10: AlexNet[5]

Surface feature classification is the next consideration from the cascade model. To reduce the confusion between true and false fire flames, it's crucial to distinguish the surfaces of fire and non-fire objects. The surface of one's clothes is considerably finer and smoother when compared with the surface of the fire flame. Dung and Ro [3] suggested to develop a fire classifier by the following equation [3]:

$$std^I = \sqrt{1/n \sum_{1}^{n-1} (I_R - I)}$$

std^I,I, and I_R are respectively the standard deviation, average, and individual intensity of candidate regions, and N denotes the number of object pixel in the image [3]. If std^I is larger than a given threshold, the candidate region will be classified as a potential fire [3]. Optical flow classification was the last feature that considered to develop a fire classifier based on whether a part of flames could possibly move in a different direction. Thus, every angles of optical flow vectors of candidate objects were calculated to classify the object as fire and move on to the next layer, the CNN classifier, once the scope of its angles is greater than a prearranged threshold.

3.3 AlexNet

The AlexNet architecture which was developed by Krizhevsky

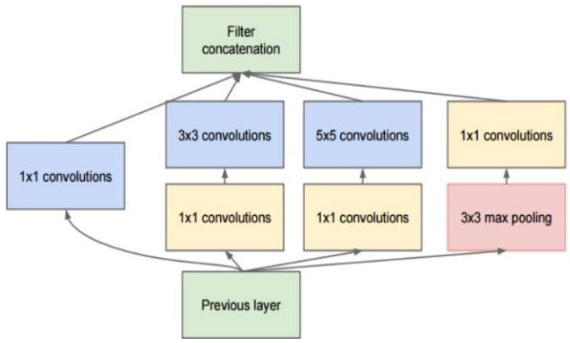

Figure 11: Inception Module[5]

in 2014 contains five convolutional layers, two fully connected layers for learning features [1]. It has become one of the most influential architecture and examined variations in CNN framework. The architecture is similar to LeNet, one of the first CNNs proposed by LeCun et al. in 1990. Their task was recognizing the handwritten digits or zip codes. However, it is much bigger and deeper compared to LeNet [1]. In general operation, different sizes of several data transformations are applied on the input data in order to generate feature maps as shown in Figure 10 [5] [1]. The feature map is a function which maps the output of one filter applied to the previous layer, and these feature maps are inserted to the next operation known as pooling where maximum feature maps are selected from them [5].

3.4 GoogleNet

GoogleNet is advanced architecture of CNN algorithm. To improve performance of CNN, they increased the number of layers in CNN. However, increase in the number of layers caused increase in the number of parameters and the chances of network overfitting also the increase in the number of layers leads to quadratic increase in the amount of computations. GoogleNet has 22 layers of architecture and uses inception module to reduce number of increasing free parameter and prevent fall in overfitting which happens when a model learns new training examples, increasing negative impacts on the performance of the model. Also, GoogleNet is deeper than other models and has less free parameter [6]. Key point of inception module is 1x1 convolution as shown on figure 11. There are two main point of 1x1 convolution. The first is to control the number of channels. Adjusting the number of channels is calculating the correlation between channels. The second is to reduce the channel of image. To be certain, the number of parameters in the 3x3 and 5x5 convolution layers are reduced by 1x1 convolution. This makes the network deeper than conventional CNN structures, but the parameters are not very large [6].

4. TRAINING CNN

Dung and Ro [3] proposed their algorithm using CaffeNet architecture of CNN, and they had training dataset of 10,000 fire images and 10,000 non-fire images, and testing dataset of 2,000 fire images and 2,000 non-fire images. Then, they ran the training for a maximum of 50,000 iterations for the optimization process. They included temporal analysis, analyzing the past image frames for a certain period, to enhance the credibility of the distinction between fire and non-fire of their proposed algorithm. They evaluated testing dataset of 15 videos including fire and non-fire objects. Furthermore, Muhammad et al. [5] explored deep learning architectures (AlexNet and GoogleNet) using larger training and testing datasets. The total number of images used in the experiments for both AlexNet and GoogleNet is 68,457. Among the total images, 62,690 frames are belongs to Dataset 1 and remainings are in Dataset 2. For the better experimental strategy, they were trained by only 20% data of dataset for training and rest of them for testing. Dataset 1 contains 31 videos, in particular, 14 fire videos and 17 normal videos without fire. The dataset has been challenged for both color and motion-based fire detection methods by fire-like objects such as red lightings and model flames or clouds. This leads to the architectures achieving more accurate fire detection outputs with challenging images. In contrast with the Dataset 1, Dataset 2 consists of 226 images not fire videos which has separated into two classes; 119 images belong to fire class and 107 images to non-fire class. It is similar to the Dataset 1 that it contains red-colored and fire-like objects in order to challenge the fire detection.

5. RESULT

Two different proposed networks of CNN algorithm were evaluated using AlexNet and GoogleNet. According to an experiment conducted by Muhammad et al. [5], their first dataset consisted of the testing dataset containing 14 videos of fire and 17 videos without fire which cover different situations in order to train for both color and motion based fire detection methods [5]. Use of the deep learning architecture named GoogleNet in this experiment resulted in a detecting accuracy of 88.41% with 0.11% for the false positives score [5]. With the fine-tuning process, the false positive rate diminished from 0.11% to 0.054% and false negative score from 5.5% to 1.5%. Using same dataset in AlexNet architecture, the results of AlexNet model resulted in an accuracy of 90.06% with false alarms rate of 9.22%. The accuracy from the AlexNet model, before fine tuning model, is quite high, however, false negatives score is 10.65% which would be problematic of fire detection shown as in Figure 12. The Dataset 2 which contains 119 images of fire and 107 images of non-fire is also challenging as it comprises red-colored and fire-like objects [5]. The results of this dataset are compared with five methods including AlexNet and GoogleNet as shown in Figure 13. The hand-crafted features based method, AlexNet, has the 0.85 Precision values, which determines the relevancy of chosen key frames, 0.92 Recall, which computes the probability of selecting a relevant frames as key frame, and the F-Measure, the average of both precision and recall, is 0.88 before running the fine-tuning process. After the tuning on the AlexNet, the precision, recall and F-measure are respectively 0.82, 0.98, and 0.89. The deep learning based method, GoogleNet, resulted in precision for 0.86, recall for 0.89 and F-measure for 0.88 before applying the fine-tuning technique. After running the fine-tuning on the GoogleNet, the precision, recall and F-measure are respectively 0.80, 0.93 and 0.86. There is no significant improvement from the results of both AlexNet and GoogleNet with the fine-tuning process. Additionally, the overall performance of Dataset 2 is not better than Dataset 1, it notes the deep learning based method is superior than hand-crafted features based fire detection methods. An-

Technique	False Positives (%)	False Negatives (%)	Accuracy (%)
Proposed after fine tuning (FT)	**0.054**	1.5	**94.43**
Proposed before FT	0.11	5.5	88.41
Muhammad et al. [2] (after FT)	9.07	2.13	94.39
Muhammad et al. [2] (before FT)	9.22	10.65	90.06
Foggia et al. [14]	11.67	**0**	93.55
De Lascio et al. [17]	13.33	0	92.86
Habibuglu et al. [15]	5.88	14.29	90.32
Rafiee et al. (RGB) [13]	41.18	7.14	74.20
Rafiee et al. (YUV) [13]	17.65	7.14	87.10
Celik et al. [11]	29.41	**0**	83.87
Chen et al. [6]	11.76	14.29	87.10

Figure 12: Comparison with different fire detection methods[5]

other research conducted by Dung and Ro [3] demonstrated the trained CNN using CaffeNet model which is a variation of AlexNet developed by Berkeley AI Research (BAIR) [3]. Briefly, the model has the five of convolution layers followed by maximum pooling layers and has next three fully connected layers, and the classification layer at the end in order to compute the classification score for each images across the innated 1,000 object types [3]. The research showed that the resulting accuracy of the trained CNN was 98% with using a big training dataset of 10,000 fire images and 10,000 non-fire images [3]. The process was operated for a maximum of 50,000 iterations [3] and it improved processing capabilities of surveillance systems for identifying the unexpected circumstances. The evaluation data included 10 videos for fire and 5 videos of non-fire objects [3]. Other previous algorithms have difficulties identifying moving objects which have similar features of fire flame or clothes etc [3]. However, this algorithm was easy to recognize the fire flame with high precision of CNNs image classifier as shown in Figure 14 [3]. Another outstanding point of this algorithm is that the processing time is less than 1 milliseconds using surface feature classification which is a part of the cascade model and including other steps. The overall time to process was about 20 milliseconds of a video [3].

Technique	Precision	Recall	F-Measure
Proposed after fine tuning (FT)	0.80	0.93	0.86
Proposed before FT	**0.86**	0.89	0.88
Muhammad et al. [2] (after FT)	0.82	**0.98**	**0.89**
Muhammad et al. [2] (before FT)	0.85	0.92	0.88
Chino et al. [30]	0.4-0.6	0.6-0.8	0.6-0.7
Rudz et al. [36]	0.6-0.7	0.4-0.5	0.5-0.6
Rossi et al. [37]	0.3-0.4	0.2-0.3	0.2-0.3
Celik et al. [11]	0.4-0.6	0.5-0.6	0.5-0.6

Figure 13: Comparison with different fire detection methods from Dataset 2 [5]

6. CONCLUSIONS

Dung and Ro [3] and Muhammad et al [5] proposed effective approaches of fire detection in surveillance videos based on convolutional neural networks and both of their algorithms are promising for accurate fire detection. Dung and Ro [3] made a cascade model and placed CNN in the lastest layer of their model to utilize the advantages of multiple classification features promising a high accuracy of smoke and flame detection. Muhammad et al [5] introduced a cost-benefit CNN architecture of fire detection for surveillance videos which is a fine-tuned version of GoogleNet architecture mainly focusing on accuracy of fire detection and computational complexity. According to the results of experiments, it is proved that the proposed architecture is more likely to be a superior architecture compared to hand-crafted features and AlexNet based fire detection. Their proposed architecture demonstrated that improvement of the flame detection accuracy. However, further studies need to work on minimizing false alarms due to the high number of false alarms still exhibited with their architecture.

Video No.	Description	Fire Detected/False Positive		
		Gunawaardena [1]	Jessica Ebert[3]	Our Algorithms
1	Outdoor day fire	Fire detected	Fire detected	Fire detected
2	Outdoor day fire	Fire detected	Fire detected	Fire detected
3	Outdoor day fire	Fire detected	Fire detected	Fire detected
4	Outdoor day fire	Fire detected	Fire detected	Fire detected
5	Human moving around fire	Fire detected	Fire detected	Fire detected
6	Outdoor day fire	Fire detected	Fire detected	Fire detected
7	Human moving around fire	Fire detected	Fire detected	Fire detected
8	Indoor night fire	Fire detected	Fire detected	Fire detected
9	Indoor night fire	Fire detected	Fire detected	Fire detected
10	Indoor night fire	Fire detected	Fire detected	Fire detected
11	Red moving human	No fire	No fire	No fire
12	Red moving human	No fire	No fire	No fire
13	Red moving human	False positive	False positive	No fire
14	Yellow moving human	No fire	No fire	No fire
15	Red moving human	False positive	False positive	No fire

Figure 14: Experimental results[3]

Acknowledgments

I sincerely thank my processor for senior seminar, Elena Machkasova, and reviewers for guidance and feedback.

7. REFERENCES

[1] S. Albawi, T. A. Mohammed, and S. Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6, Aug 2017.

[2] N. Aloysius and M. Geetha. A review on deep convolutional neural networks. In *2017 International Conference on Communication and Signal Processing (ICCSP)*, pages 0588–0592, April 2017.

[3] N. M. Dung and S. Ro. Algorithm for fire detection using a camera surveillance system. In *Proceedings of the 2018 International Conference on Image and Graphics Processing*, ICIGP 2018, pages 38–42, New York, NY, USA, 2018. ACM.

[4] K. Muhammad, J. Ahmad, Z. Lv, P. Bellavista, P. Yang, and S. W. Baik. Efficient deep cnn-based fire detection and localization in video surveillance applications. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–16, 2018.

[5] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik. Convolutional neural networks based fire detection in surveillance videos. *IEEE Access*, 6:18174–18183, 2018.

[6] H. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, 35(5):1285–1298, May 2016.

[7] G.-Y. Son, Marshall, J. Par k, and D.-H. Lee. An analysis of parameters of convolutional neural network for fire detection. In *Proceedings of the 2Nd International Conference on Software Engineering and Information Management*, ICSIM 2019, pages 21–24, New York, NY, USA, 2019. ACM.

[8] Wikipedia. Artificial neural network — Wikipedia, the free encyclopedia, 2015. [Online; accessed 24-April-2019].

[9] Wikipedia. Convolutional neural network — Wikipedia, the free encyclopedia, 2015. [Online; accessed 24-April-2019].

[10] Wikipedia. Deep learning — Wikipedia, the free encyclopedia, 2015. [Online; accessed 24-April-2019].