



e-ISSN: 2319-8753 | p-ISSN: 2347-6710

IJRSET

International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Special Issue 3, December 2021

ISSN INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

Impact Factor: 7.569

9940 572 462

6381 907 438

ijirset@gmail.com

www.ijirset.com

Convolutional Neural Network Based SqueezeNet Framework for the Detection of Fire Disaster and Localization in Video Surveillance Systems

Saranya Mukunadan C, Shanid Malayil Sir,

Student, Royal College of Engineering Technology, APJ Abdul Kalam Technological University, Kerala, India

HOD, Department of CSE, Royal College of Engineering, Kerala, India

ABSTRACT: The recent advances in embedded processing have enabled the vision based systems to detect fire during surveillance using convolutional neural networks (CNNs). However, such methods generally need more computational time and memory, restricting its implementation in surveillance networks. In this research paper proposes a cost-effective fire detection CNN architecture for surveillance videos. The model is inspired from SqueezeNet architecture, considering its reasonable computational complexity and suitability for the intended problem compared to other computationally expensive networks such as AlexNet. To balance the efficiency and accuracy, the model is fine-tuned considering the nature of the target problem and fire data. Experimental results on benchmark fire datasets reveal the effectiveness of the proposed framework and validate its suitability for fire detection in CCTV surveillance systems compared to state-of-the-art methods.

KEYWORDS: Convolutional neural networks (CNNs), deep learning, fire detection, fire disaster, fire localization, image classification, surveillance networks.

I. INTRODUCTION

Emergency situations like floods, earthquakes and fires pose a big threat to public health and safety, property and environment. Fire related disasters are the most common type of Emergency situation which requires thorough analysis of the situation required for a quick and precise response. The first step involved in this process is to detect fire in the environment as quickly and accurately as possible. Fire Detection in most places like temperature detectors, smoke detectors, thermal cameras etc. which is expensive and not available to all. But, after the advent of advanced image processing and computer vision techniques, detection of fire may not require any equipment other than cameras. Due to this expeditious development in vision-based fire detection models, there is a particular inclination towards replacing the traditional fire detection tools with vision-based models. These models have many advantages over their hardware based counterparts like accuracy, more detailed view of the situation, less prone to errors, robustness towards the environment, considerably lower cost and the ability to work on existing camera surveillance systems.

Detecting fire in images using image processing and computer vision techniques has gained a lot of attention from researchers during the past few years. Indeed, with sufficient accuracy, such systems may outperform traditional fire detection equipment. One of the most promising techniques used in this area is Convolutional Neural Networks (CNNs). However, the previous research on fire detection with CNNs has only been evaluated on balanced datasets, which may give misleading information on real-world performance, where fire is a rare event. In this paper proposes to use deeper Convolutional Neural Networks for fire detection in images, and enhancing these with fine tuning based on a fully connected layer because CNN performs relatively poorly when evaluated on the more realistically balanced benchmark dataset. The available literature dictates that flame detection using visible light camera is the generally used fire detection method, which has three categories including pixel-level, blob-level, and patch-level methods. The pixel-level methods are fast due to usage of pixel-wise features such as colors and stickers, however, their performance is not attractive as such methods can be easily biased. Compared to pixel-level methods, blob-level flame detection methods show better performance as such methods consider blob-level candidates for features extraction to detect flame. The

major problem with such methods is the difficulty in training their classifiers due to numerous shapes of fire blobs. Patch-level algorithms are developed to improve the performance of previous two categories of flame detection algorithms, however, such methods result in many outliers, affecting their accuracy.

To improve the accuracy, researchers attempted to explore color and motion features for flame detection. For instance, Chen et al. [3] investigated the dynamic behavior and irregularity of flames in both RGB and HSI color spaces for fire detection. Since, their method considers the frame difference during prediction, hence, it fails to differentiate real fire from fire-like moving outliers and objects. Besides RGB and HSI color models, Marbach et al explored YUV color model in combination with motion features for prediction of fire and non-fire pixels. A similar method is proposed by Töreyn et al. [2] by investigating temporal and spatial wavelet analysis, however, the excessive use of parameters by this method limits its usefulness. Another method is presented by Han and Lee by comparing the video frames and their color features for flame detection in tunnels. Continuing the investigation of color models, Celik and Demirel used YCbCr with specific rules of separating chrominance component from luminance. The method has potential to detect flames with good accuracy but at small distance and larger size of fire only. Considering these limitations, Borges and Izquierdo attempted to detect fire using a multimodal framework consisting of color, skewness, and roughness features and Bayes classifier.

In continuation with Borges and Izquierdo work, multi-resolution 2D wavelets combined with energy and shape are explored by Rafiee et al. [23] in an attempt to reduce false warnings, however, the false fire alarms still remained significant due to movement of rigid body objects in the scene. An improved version of this approach is presented in [24] using YUC instead of RGB color model, providing better results than [23]. Summarizing the color based methods, it can be noted that such methods are sensitive to brightness and shadows. As a result, the number of false warnings produced by these methods is high. The presented method uses optical flow information and behavior of flame to intelligently extract a feature vector based on which flame and moving rigid objects can be differentiated. To further improve the accuracy, Foggia et al. [14] combined shape, color, and motion properties, resulting in a multi-expert framework for real-time flame detection. Although, the method dominated state-of-the-art flame detection algorithms, yet there is still space for improvement. In addition, the false alarming rate is still high and can be further reduced. From the aforementioned literature, it is observed that fire detection accuracy has inverse relationship to computational complexity. With this motivation, there is a need to develop fire detection algorithms with less computational cost and false warnings, and higher accuracy. Considering the above motivation, we extensively studied convolutional neural networks (CNNs) for flame detection at early stages in CCTV surveillance videos. The main contributions of this article are summarized as follows:

- Exploring Deep learning architectures for early fire detection in closed circuit television (CCTV) surveillance networks for both indoor and outdoor environments is a less time consuming effort than traditional fire detection.
- The proposed Fire detection using CNN architecture improves fire detection accuracy and reduces false alarms, compared to state-of-the-art methods. The algorithm suggested in this paper can play an important role in detection of fire at its early stage to minimize damage.
- We propose an architecture similar to SqueezeNet model, to fine-tune our model and to reduce the size of the model that helps in detecting fire. An extra space of 235MB is saved by reducing the size of the model from 238MB to 3MB.
- Using Squeezenet architecture helps in minimizing the cost and making its implementation feasible and efficient.
- A feature map selection algorithm is developed which can intelligently choose appropriate feature maps from the convolutional layers of the trained CNN, which are sensitive to fire regions. These feature map algorithms allow a more accurate segmentation of fire compared to other handcrafted methods.
- The segmentation information can be further analyzed to assess the essential characteristics of the fire, for instance its growth rate. Using this approach, the severity of the fire and/or its burning degree can also be determined.
- Another novel characteristic of our system is the ability to identify the object which is on fire, using a pre-trained model trained on 1000 classes of objects in the ImageNet dataset. This enables our approach to determine whether the fire is in a car, a house, a forest or any other object. Using this semantic information, firefighters can prioritize their targets by primarily focusing on regions with the strongest fire.

II. PROPOSED FRAMEWORK

Majority of the research since the last decade is focused on traditional features extraction methods for flame detection. The major issues with such methods is their time

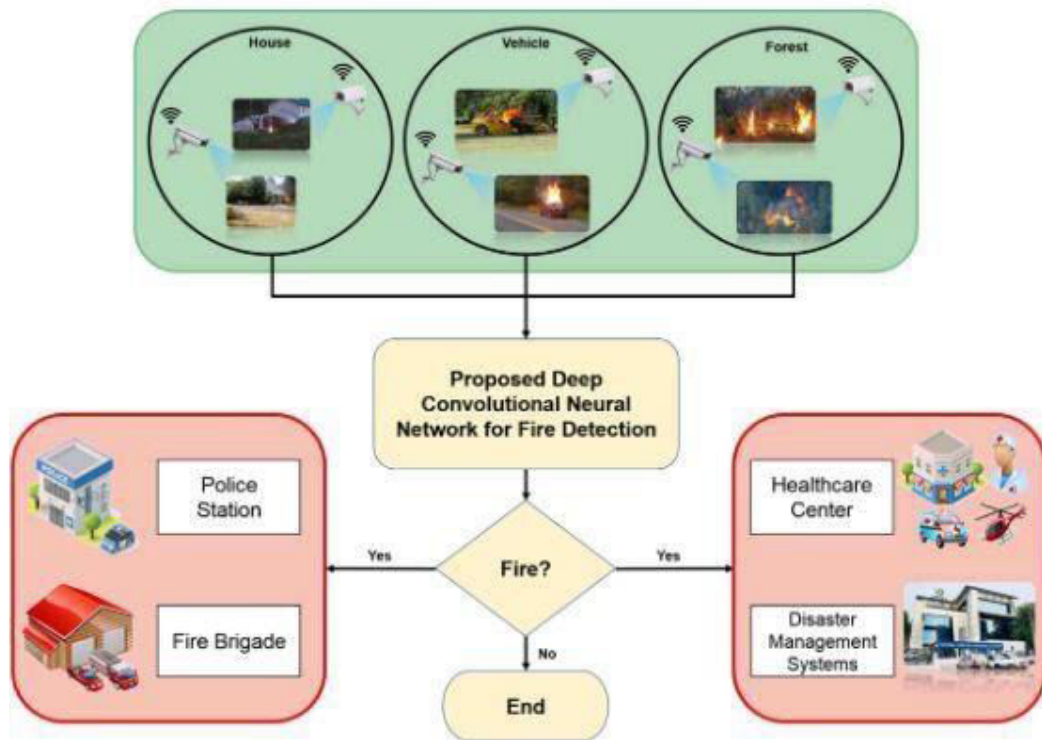


Figure 1. Overview of the proposed system for fire detection using a deep CNN.

consuming process of features engineering and their low performance for flame detection. Such methods also generate high number of false alarms especially in surveillance with shadows, varying lightings, and fire-colored objects. It is particularly challenging to detect a fire at an early stage in scenes with changing lighting conditions, shadows, and fire-like objects; conventional low-level feature-based methods generate a high rate of false alarms and have low detection accuracy. To overcome these issues, we investigate deep learning models for possible fire detection at early stages during surveillance. To achieve this goal, we explored deep CNNs and devised a fine-tuned architecture for early fire detection during surveillance. A systematic diagram of our framework is given in Fig 1

III. CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network is a type of neural network that is most often applied to image processing problems. A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The preprocessing required in ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNet have the ability to learn these filters/characteristics. A regular neural network has an input layer, hidden layers and output layer Fig 2.

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used. The input layer accepts inputs in different forms while hidden layers perform calculations on these inputs. The output layer then delivers the outcome of the calculations and extractions. Each of these layers contain neurons

that are connected to neurons in the previous layer and each neuron has its own weight. This means no assumptions about the data being fed into the network. This feature but not usually works with images or languages. Convolutional neural network work differently as they treat data as spatial. Instead of neurons being connected to every neuron in the previous layer, they are instead only connected to neurons close to it and all have the same weight. The simplification in the connections means the network upholds the spatial aspect of the data set. This property of convolutional neural network make it unique from other neural network. It means the network doesn't

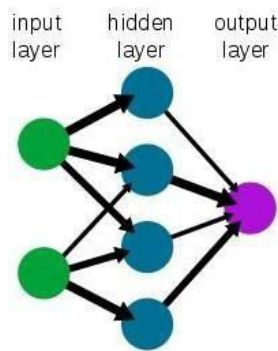


Figure 2. Regular neural network

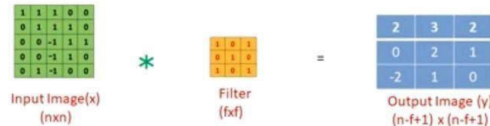


Figure 3. Image matrix multiplies kernel or filter matrix

think an eye is all over the image. The word convolutional refers to the filtering process that happens in this type of network. Consider a complex image, a convolutional neural network simplifies it, so it can be better processed and understood.

IV. CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE



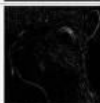


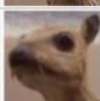

A typical CNN architecture is consists of three well- known processing layers

1. **Convolution Layer:** The primary purpose of convolution is to extract features from input image. Convolution preserves the spatial relationship between input by learning input features. A convolution sweeps the window through images then calculates its input and filter dot product pixel values. This allows convolution to emphasize the relevant features. The convolution layer produces output by simply convolving input image with a filter where x is a input image which is actually representing the input image with $n \times n$ matrix remember the computer consider 0 1 in grayscale image, when it is RGB then input image values varies from 0 to 255. Fig 3 shows an $n \times n$ matrix input image with $f \times f$ filter. Here n represents size of input image and f represents size of filter. Consider a 5×5 image matrix whose image pixel values are 0, 1 and filter matrix 3×3 as shown in fig.3. Then the convolution of 5×5 image matrix multiplies with 3×3 filter matrix which is called "Feature Map" as output shown in below in fig 4. Convolution of an

Organized by

Royal College of Engineering and Technology, Thrissur, Kerala, India

Figure 4. Convolutional Layer Operation - Out- put matrix

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

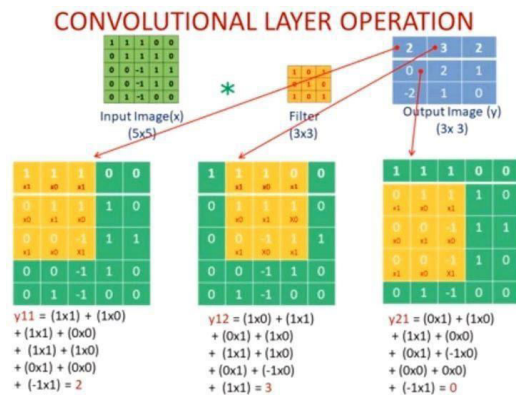


Figure 5. Some common filters

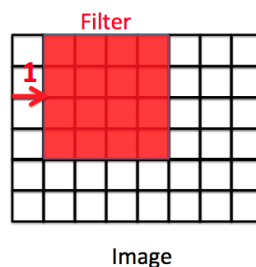


Figure 6. If Stride = 1, the filter moves 1 pixel

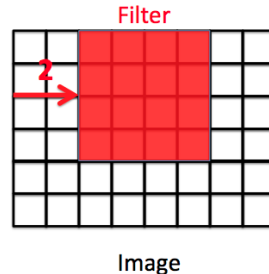


Figure 7. If Stride = 2, the filter moves 1 pixel

image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example fig 5 shows various convolution image after applying different types of filters (Kernels). A Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. Let's look at an example. The red square is a filter. The computer is going to use this filter to scan the images fig 6 & fig 7. we can define how far the filter moves from one position to the next position by "stride".

2. **Pooling Layer:** A pooling layer is a new layer added after the convolutional layer. Specifically, after a non linearity (e.g. ReLU) has been applied to the feature maps output by a convolutional layer; for example the layers in a model may look as follows:

- Input Image
- Convolutional Layer
- Non linearity
- Pooling Layer

The addition of a pooling layer after the convolutional layer is a common pattern used for ordering layers within a convolutional neural network that may be repeated one or more times in a given model. The pooling layer operates upon each feature map separately to create a new set of the same number of pooled feature maps. Pooling involves selecting a pooling operation, much like a filter to be applied to feature maps. The size of the pooling operation or filter is smaller than the size of the feature map; specifically, it is almost always 2x2 pixels applied with a stride of 2 pixels. This means that the pooling layer will always reduce the size of each feature map by a factor of 2, e.g. each dimension is halved, reducing the number of pixels or values in each feature map to one quarter the size. For example, a pooling layer applied to a feature map of 6x6 (36 pixels) will result in an output pooled feature map of 3x3 (9 pixels). A pooling layer, which is used for the selection of maximum activation considering a small neighborhood of feature maps received from the previous convolution layer; the goal of this layer is to achieve translation invariance to some extent and dimensionality reduction. Pooling also called down sampling which reduces the dimensionality of each map but retains important information. There are three types of pooling

- a. Max Pooling
- b. Average Pooling
- c. Sum Pooling

CNN uses max pooling to replace output with a max summary to reduce data size and processing time. This allows you to determine features that produce the highest impact and reduces the risk of overfitting. Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling. Max pooling takes two hyper parameters: stride and size. The stride will determine the skip of value pools while the size will determine how big the value pools in every skip. The result of using a pooling layer and creating down sampled or pooled feature maps is a summarized version of the features detected in the input. They are useful as small changes in the location of the feature in the input detected by the convolutional layer will result in a pooled feature map with the feature in the same location. This capability added by pooling is called the model's invariance to local translation.

3. **FC Layer:** The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network. The role of flatten layer is to convert matrix into vector. The input to flatten layer is the output of max pooling layer. Fig 9. Next is fully connected layer, the output of flatten layer is taken as input for fully connected

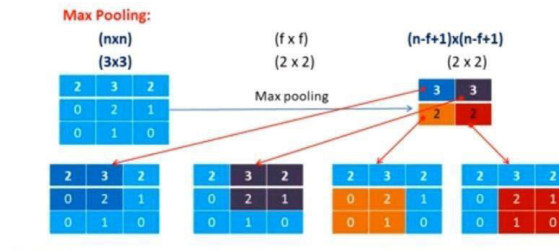


Figure 8. Pooling Layer

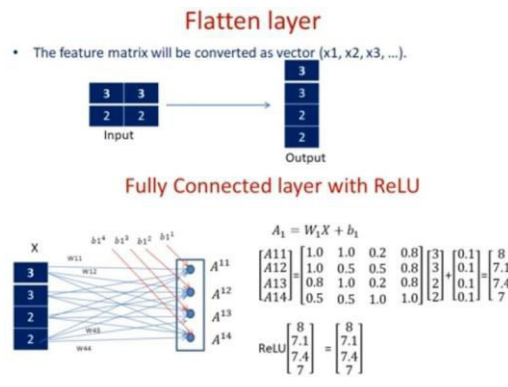


Figure 9. FC layer

layer. In fig the fully connected layer has four neurons each receives bias namely b_{11} , b_{12} , b_{13} , b_{14} and each cell or neurons receive input from all the four input after multiplying with corresponding bias. This weight and bias have to be initialized before running the program's in order to calculate the output of fully connected layer we have to use this formula $A = WX + B$ Where w is weight matrix of a layer is input for a layer, b is bias of same layer. A fully connected layer which models high-level information from the input data and constructs its global representation. This layer follows numerous stacks of convolution and pooling layers, thus resulting in a high level representation of the input data.

These layers are arranged in a hierarchical architecture such that the output of one layer acts as the input of the next layer to form a complete CNN architecture shown in fig 10. During the training phase, the weights of all neurons in convolutional kernels and fully connected layers are adjusted and learned. These weights model the representative characteristics of the input training data, and in turn can perform the target classification. We use a model with an architecture similar to that of SqueezeNet [10], modified in accordance with our target problem. The original model was trained on the ImageNet dataset and is capable of classifying 1000 different objects. In our case, however, we used this architecture to detect fire and non fire images. This was achieved by reducing the number of neurons in the final layer from 1000 to 2. By keeping the rest of the architecture similar to the original, we aimed to reuse the parameters to solve the fire detection problem more effectively.

V. OVERVIEW OF STEPS IN CNN

1. Provide input image into convolution layer
2. Choose parameters, apply filters with strides, padding if requires. Perform convolution on the image and apply ReLU activation to the matrix.
3. Perform pooling to reduce dimensionality size
4. Add as many convolutional layers until satisfied
5. Flatten the output and feed into a fully connected layer (FC Layer)
6. Output the class using an activation function (Logistic Regression with cost functions) and classifies images.

VI. SQUEEZENET ARCHITECTURE

SqueezeNet is a smaller network that was designed as a more compact replacement for AlexNet. It has almost 50x fewer parameters than AlexNet, yet it performs 3x faster. This architecture shown in fig 11 was proposed by researchers at Deep-Scale, The University of California, Berkeley, and Stanford University in the year 2016. It was first published in their paper titled SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. SqueezeNet is a convolutional neural network that employs design strategies to reduce the number of parameters, notably with the use of fire modules that "squeeze" parameters using 1x1 convolutions. There are several motivational reasons for the selection of SqueezeNet architecture, such as a lower communication cost between different servers in the case of distributed training, a higher feasibility of deployment on FPGAs, application-specific integrated circuits, and other hardware architectures with memory constraints and lower bandwidth. SqueezeNet is a small CNN architecture called that achieves AlexNet- level accuracy on ImageNet with 50x fewer parameters. Three advantages of small CNN architectures:

1. Require less communication across servers during distributed training.
2. Require less bandwidth to export a new model from the cloud.
3. More feasible to deploy on customized hardware with limited memory.

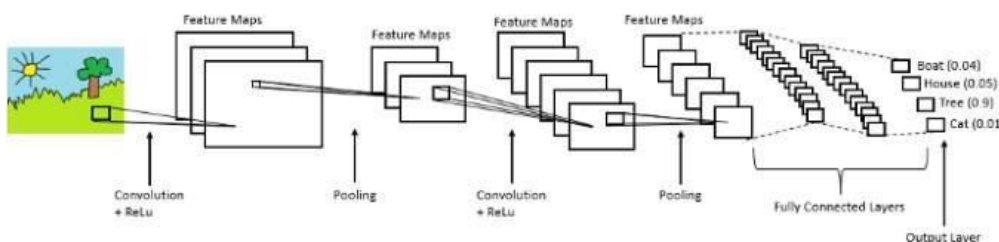


Figure 10. Complete CNN architecture.

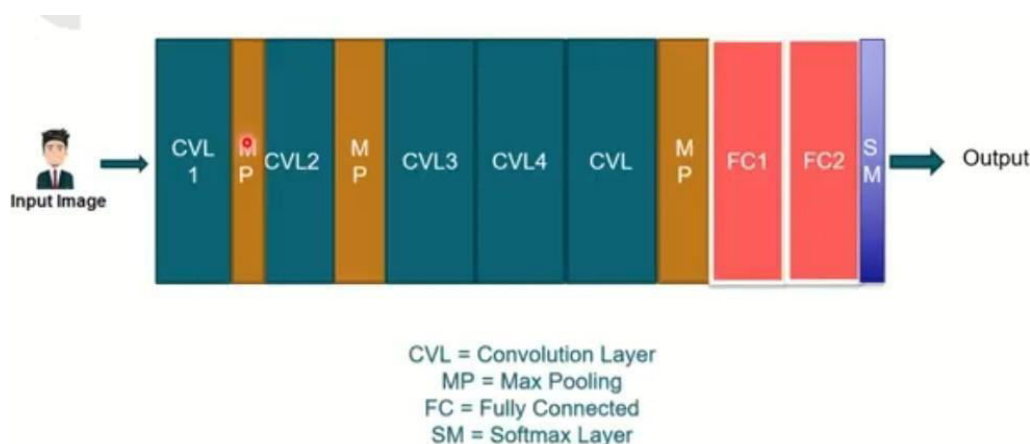


Figure 11. Easy Representation of SqueezeNet architecture.

1. Architectural Design Strategies

1. **Strategy 1. Make the network smaller by replacing 3x3 filters with 1x1 filters.** This strategy reduces the number of parameters 9x by replacing a bunch of 3x3 filters with 1x1 filters. At first this seemed really confusing to me. By moving 1x1 filters across an image I would think that each filter has less information to look at and would thus perform more poorly, however that doesn't seem to be the case! Typically a larger 3x3 convolution filter captures spatial information of pixels close to each other. On the other hand, 1x1 convolutional filters zero in on a single pixel and capture relationships amongst its channels as opposed to neighboring pixels.

2. **Strategy 2. Reduce the number of inputs for the remaining 3x3 filters.** This strategy reduces the number of parameters by basically just using fewer filters. The systematic way this is done is by feeding "squeeze"

layers into what they term “expand” layers as shown in fig 3.13: “squeeze” layers are convolution layers that are made up of only 1x1 filters and “expand” layers are convolution layers with a mix of 1x1 and 3x3 filters. By reducing the number of filters in the “squeeze” layer feeding into the “expand” layer, they are reducing the number of connections entering these 3x3 filters thus reducing the total number of parameters. The authors of this paper call this specific architecture the “fire module” and it serves as the basic building block for the SqueezeNet architecture.

3. **Strategy 3. Downsample late in the network so that convolution layers have large activation maps** the intuition is that large activation maps (due to de-layer down sampling) can lead to higher classification accuracy. Decreasing the stride with later convolution layers and thus creating a larger activation/feature map later in the network, classification accuracy actually increases. Having larger activation maps near the end of the network is in stark contrast to networks like VGG where activation maps get smaller as you get closer to the end of a network.

2. Fire Module

SqueezeNet takes advantage of the aforementioned “fire module” and chains a bunch of these modules together to

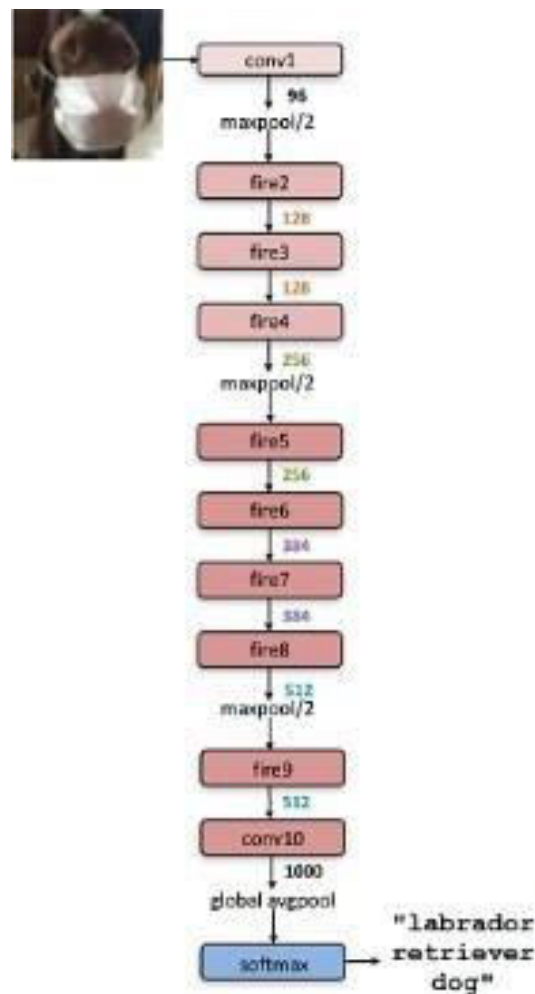


Figure 12. Complete SqueezeNet architecture.

Arrive at a smaller model. The authors of the paper use the term “fire module” to describe a squeeze layer and an expand layer together discussed in next section. An input image is sent into a standalone convolutional layer. This layer is followed by 8 “fire modules” which are named “fire2-9”, according to Strategy One above. The illustration of the resulting SqueezeNet is shown in figure 12. A fire module is comprised of a squeeze convolution layer (which has only 1×1 filters), feeding into an expand layer that has a mix of 1×1 and 3×3 convolution filters. A fire module is the building block used in the SqueezeNet employs Strategies 1, 2, and 3 comprised of squeeze layers which have only 1x1 filters (strategy 1) comprised of expand layers which have a mix of 1x1 and 3x3 convolution filters number of Filters in squeeze layer must be less than the expand layer (strategy 2) as shown in figure 13.

4. Proposed CNN Architecture

We use a model with an architecture similar to that of SqueezeNet [10], modified in accordance with our target problem. It had a total of five convolution layers, three pooling layers, and two fully connected layers and a soft-max layer as shown in Figure 10. The original model was trained on the ImageNet dataset and is capable of classifying 1000 different objects. In our case, however, we used this architecture to detect fire and nonfire images. This was achieved by reducing the number of neurons in

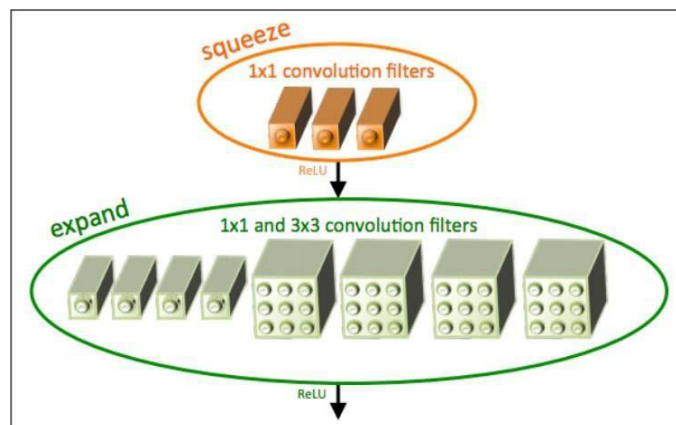


Figure 13. Fire module

The final layer from 1000 to 2. By keeping the rest of the architecture similar to the original, we aimed to reuse the parameters to solve the fire detection problem more effectively. There are several motivational reasons for this selection, such as a lower communication cost between different servers in the case of distributed training, a higher feasibility of deployment on FPGAs, application-specific integrated circuits, and other hardware architectures with memory constraints and lower bandwidth. The model consists of two regular convolutional layers, three max pooling layers, one average pooling layer, and eight modules called “fire modules.” The input of the model is color images with dimensions of 224×224×3 pixels. In the first convolution layer, 64 filters of size 3×3 are applied to the input image, generating 64 feature maps. The maximum activations of these 64 features maps are selected by the first max pooling layer with a stride of two pixels, using a neighborhood of 3×3 pixels. This reduces the size of the feature maps by factor of two, thus retaining the most useful information while discarding the less important details. Next, we use two fire modules of 128 filters, followed by another fire module of 256 filters. Each fire module involves two further convolutions, squeezing, and expansion. Since each module consists of multiple filter resolutions and there is no native support for such convolution layers in the Cafe framework [36], an expansion layer was introduced, with two separate convolution layers in each fire module. The first convolution layer contains 1×1 filters, while the second layer consists of 3×3 filters. The output of these two layers is concatenated in the channel dimension. Following the three fire modules, there is another max pooling layer which operates in the same way as the first max pooling layer. Following the last fire module (Fire9) of 512 filters, we modify the convolution layer according to the problem of interest by reducing the number of classes to two [M = 2 (fire and normal)]. The output of this layer is passed to the average pooling layer, and result of this

Layer is fed directly into the Softmax classifier to calculate the probabilities of the two target classes.

5. Fire detection and localization

Deep CNN based fire detection and localization explains the process of fire detection and its localization using the proposed deep CNN. CNN architectures is capable of learning strong features from raw data, still some efforts are required to train the appropriate model considering the quality and quantity of the available data and the nature of the target problem. Our proposed model is trained with various models with different parameter settings, and following the fine-tuning process obtained an optimal model which can detect fire from a large distance and at a small scale, under varying conditions, and in both indoor and outdoor scenarios. The proposed deep CNN was the avoidance of preprocessing and features engineering, which are required by traditional fire detection algorithms. To detect an image with fire or non-fire, it is fed forward through the deep CNN architecture, which assigns a label of “fire” or “normal” to the input image. This label is assigned based on probability scores computed by the network. The higher probability score is taken to be the final class label of the input image. A set of sample images with their predicted class labels and probability scores is given in Figure 14. To localize a fire in a given sample input image we employ the framework given in Figure 15. A prediction is obtained first from the proposed deep CNN architecture. After prediction the input image is classified into nonfire if image consist of not fire, no further action is performed; in the case of fire image, further processing is undertaken that is fire localization is performed. Two algorithms are considered in fire localization as given in Algorithms 1 and 2. After analyzing all the feature maps extracted from the different layers of our proposed CNN using Algorithm 1, feature maps 8, 26, and 32 of the “Fire2/Concat” layer



Figure 14. Prediction scores for a set of query images using the proposed deep CNN. (a) Fire: 100%, normal: 0.0%. (b) Fire: 99.35%, normal: 0.47%. (c) Fire: 99.98%, normal: 0.02%. (d) Fire: 99.46%, normal: 0.54%. (e) Fire: 0.95%, normal: 99.05%. (f) Fire: 14.46%, normal: 85.54%. (g) Fire: 40.91%, normal: 59.09%. (h) Fire: 13.56%, normal: 86.44%.

found to be sensitive to fire regions and to be appropriate for fire localization. We therefore fused these three feature maps and applied binarization to segment the fire. A set of sample fire images with their segmented regions is given in Figure 16. The segmented fire is used for two further purposes: 1) determining the severity level/burning degree of the scene under observation and 2) finding the zone of influence (ZOI) from the input fire image. The burning degree can be determined from the number of pixels in the segmented fire. The ZOI can be calculated by subtracting the segmented fire regions from the original input image. The resultant ZOI image is then passed from the original SqueezeNet model [10], which predicts its label from 1000 objects. The object information can be used to determine the situation in the scene, such as a fire in a house, a forest, or a vehicle. This information, along with the severity of the fire, can be reported to the fire brigade to take appropriate action.

VI. RESULTS AND DISCUSSIONS

This section explains in detail the experiments conducted to evaluate the performance of the proposed framework. The experiments are focused on two datasets: Fog-gia’s video dataset [14] and Chino’s dataset [15]. The first dataset consists of 31 videos with both indoor and outdoor environments, of which 14 videos contain fire and the remaining 17 videos are without fire. The reasons for selecting this dataset include its large number of videos

Captured in different scenes in indoor and outdoor environments as well as its challenges. For instance, the last 17 videos contain fire-like objects and situations, which can be predicted as fire, making the classification more difficult. To this end, color-based methods may fail to differentiate between real fire and scenes with red color objects. Similarly, motion-based techniques may wrongly classify a scene with mountains containing smoke, cloud, or fog. These compositions make the dataset more challenging, enabling us to stress our framework and investigate its performance in various situations of the real environment. The detailed information about this dataset and a set of images from it are shown in Table 1 and figure 17. The results achieved based on this dataset and its comparison with state-of-the-art fire detection methods are shown in Table 2.

Fig. 17 shows sample images from selected videos of the first dataset of 31 videos. Fig. 11 (i)–11(vi) represent the frames containing fire, while Fig. 11 (VI)–11 (xii) show images containing no fire. It can be noted from the given images that the dataset contains frames belonging to both indoor (Fire13) and outdoor (Fire1, Fire2 etc.) environments. The images also illustrate that some videos contain a large amount of fire, such as Fire3, and some have very little fire, such as Fire13 and Fire14. Another challenge introduced in the dataset is the distance of the fire from the camera. For instance, Fire2 video contains a very small fire at a larger distance. On the other hand, Fire13 video indicates a small fire but at a comparatively small distance.

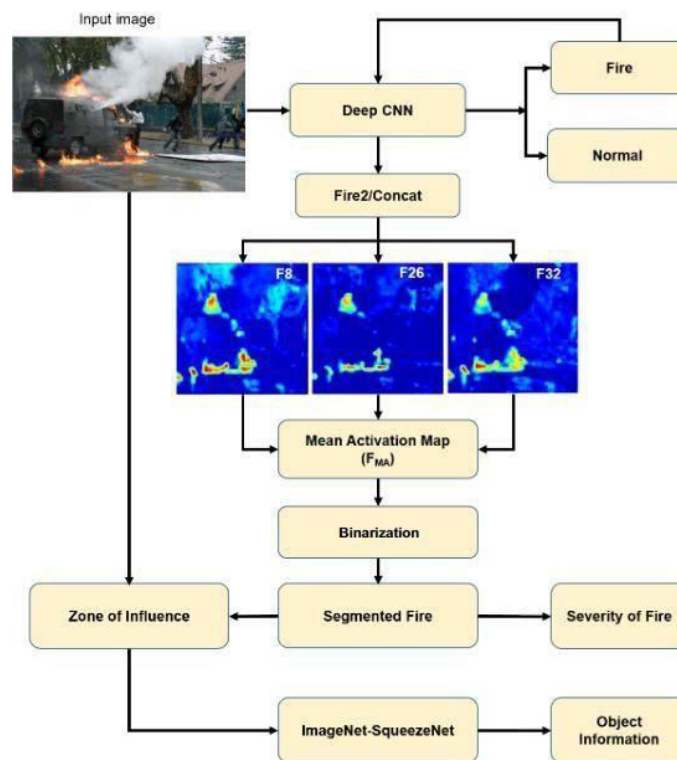


Figure 15. Fire localization using the proposed deep CNN.

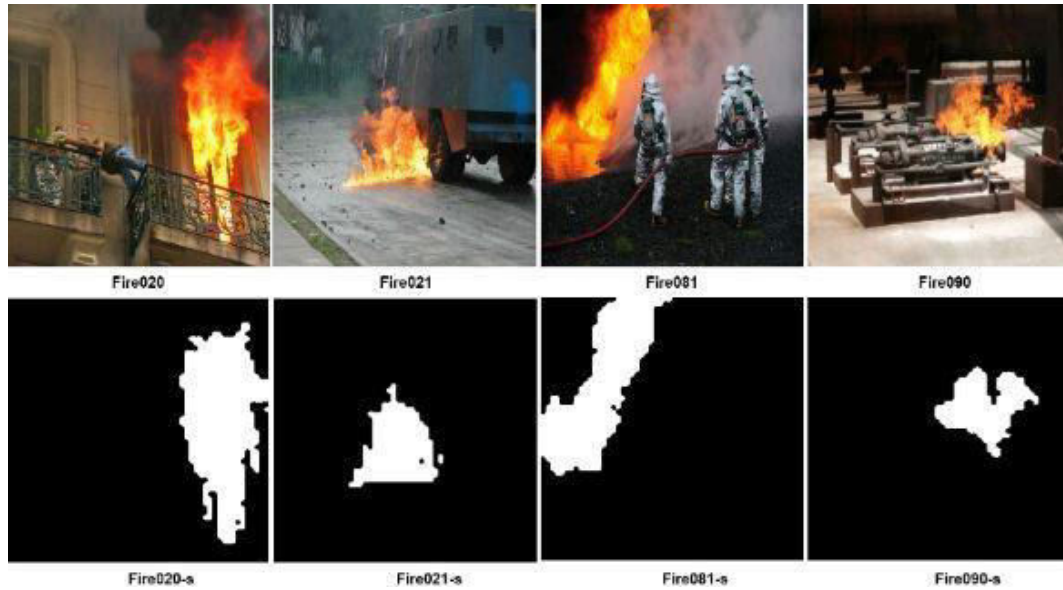


Figure 16. Sample images and the corresponding localized fire regions using our approach. The first row shows the original images, while the second row shows the localized fire regions.

Besides this, there are red-colored objects and grounds such as a signboard (Fire14) and reddish grass (Fire6) in many videos, making the dataset very challenging for fire detection methods. The proposed work is compared with

other related methods in Table 2. The existing methods for comparison are selected carefully. Considering the underlying dataset, year of publication, and features used. From the results given in Table 2, the best method in terms of false positives is [22], but its false negatives are greater than other methods except for [3]. In addition, its accuracy is 90.32%, which is less than two existing methods [14] and [21] as well as the proposed work. The work of [13] is good compared to other methods, but the false positives are still 11.67% and there is still room for improvement in both accuracy and false positives. The proposed work, inspired from deep features, reported further improvement by increasing the accuracy from 93.55% to 94.39% and reducing the false positives from 11.67% to 9.07%. Although, our work also resulted in false negatives of 2.13%, it still maintained a better balance between accuracy, false positives, and false negatives, making our method more suitable for early fire detection, which is of paramount interest to disaster management systems.

Organized by

Royal College of Engineering and Technology, Thrissur, Kerala, India

Algorithm 1 Feature Map Selection Algorithm for Localization**Input:** Training samples (TS), ground truth (GT), and the proposed deep CNN model (CNN-M)

1. Forward propagate TS through CNN-M
2. Select the feature maps F_N from layer L of CNN-M
3. Resize GT and F_N to 256×256 pixels
4. Compute mean activations map F_{MAi} for F_N
5. Binarize each feature map F_i as follows:

$$F(x, y)_{bin(i)} = \begin{cases} 1, & F(x, y)_i > F_{MA(i)} \\ 0, & \text{Otherwise} \end{cases}$$

6. Calculate the hamming distance HD_i between GT and each feature map $F_{bin(i)}$ as follows:

$$HD_i = |F_{bin(i)} - GT|$$

This results in $TS \times F_N$ hamming distances

7. Calculate the sum of all resultant hamming distances, and shortlist the minimum hamming distances using threshold T
8. Select appropriate feature maps according to the short-listed hamming distances

Output: Feature maps sensitive to fire**Algorithm 2** Fire Localization Algorithm**Input:** Image I of the video sequence and the proposed deep CNN model (CNN-M)

1. Select a frame from the video sequence and forward propagate it through CNN-M
2. **IF** predicted label = non-fire **THEN**
No action
- ELSE**
 - a) Extract feature maps 8, 26, and 32 (F_8, F_{26}, F_{32}) from the "Fire2/Concat" layer of CNN-M
 - b) Calculate mean activations map (F_{MA}) for F_8, F_{26} , and F_{32}
 - c) Apply binarization on F_{MA} through threshold T as follows:

$$F_{Localize} = \begin{cases} 1, & F_{MA} > T \\ 0, & \text{Otherwise} \end{cases}$$

- d) Segment fire regions from F_{MA}

END**Output:** Binary image with segmented fire $I_{localize}$

The second dataset [18] is comparatively small but very challenging. The total number of images in this dataset is 226, out of which 119 images contain fire while the remaining 107 are fire-like images containing sunsets, fire-like lights, and sunlight coming through windows etc. A set of selected images from this dataset are shown in Fig. 18. For better evaluation of the performance, the results for this dataset are collected using another set of metrics including precision [19], recall, and F-measure [20]. The results achieved by our method using this dataset are reported and compared with existing methods in Table 3. By using deep features and fine tuning our fire detection model, we successfully outperformed the state-of-the-art methods by achieving the highest score of precision 0.82, recall 0.98, and F-measure 0.89, validating the effectiveness of our early fire detection method.

The performance of our approach is evaluated in terms of fire localization and understanding of the scene under observation. True positive and false positive rates were computed to evaluate the performance of fire localization. The feature maps used to localize fire were smaller than the ground truth images, and were therefore resized to match the size of the ground truth images. We then computed the number of overlapping fire pixels in the detection maps and ground truth images, and used these as true positives. Similarly, we also determined the number of non-overlapping fire pixels in the detection maps and interpreted these as false positives. One further reason for using SqueezeNet was the ability of the model to give larger sizes for the feature maps by using smaller kernels and avoiding pooling layers. This allowed us to perform more accurate localization when the feature maps were resized to match the ground truth images.



National Conference on Emerging Technologies [NCET' 2021]

Organized by

Royal College of Engineering and Technology, Thrissur, Kerala, India

Our system selects suitable features which are sensitive to fire using Algorithm 1, and localizes

TABLE I
DETAILS OF DATASET I

Video Name	Resolution	Frames	Frame Rate	Modality	Description
Fire1	320×240	705	15	Fire	Fire in a bucket with person walking around it
Fire2	320×240	116	29	Fire	Fire at a comparatively long distance from the camera in a bucket
Fire3	400×256	255	15	Fire	A large forest fire
Fire4	400×256	240	15	Fire	Same description as Fire3
Fire5	400×256	195	15	Fire	Same description as Fire3
Fire6	320×240	1200	10	Fire	Fire on the ground with red color
Fire7	400×256	195	15	Fire	Same description as Fire3
Fire8	400×256	240	15	Fire	Same description as Fire3
Fire9	400×256	240	15	Fire	Same description as Fire3
Fire10	400×256	210	15	Fire	Same description as Fire3
Fire11	400×256	210	15	Fire	Same description as Fire3
Fire12	400×256	210	15	Fire	Same description as Fire3
Fire13	320×240	1650	25	Fire	An indoor environment with fire in a bucket
Fire14	320×240	5535	15	Fire	A paper box, inside which a fire is burning
Fire15	320×240	240	15	Normal	Smoke visible from a closed window with the appearance of a red reflection of the sun on the glass
Fire16	320×240	900	10	Normal	Smoke from a pot near a red dust bin.
Fire17	320×240	1725	25	Normal	Smoke on the ground with nearby trees and moving vehicles
Fire18	352×288	600	10	Normal	Smoke on the hills, far from the camera
Fire19	320×240	630	10	Normal	Smoke on red-colored ground
Fire20	320×240	5958	9	Normal	Smoke on the hills, with nearby red buildings
Fire21	720×480	80	10	Normal	Smoke at a larger distance behind trees
Fire22	480×272	22500	25	Normal	Smoke behind hills in front of UOS
Fire23	720×576	6097	7	Normal	Smoke above hills
Fire24	320×240	342	10	Normal	Smoke in a room
Fire25	352×288	140	10	Normal	Smoke at a larger distance from a camera in a city
Fire26	720×576	847	7	Normal	Same description as Fire24
Fire27	320×240	1400	10	Normal	Same description as Fire19
Fire28	352×288	6025	25	Normal	Same description as Fire18
Fire29	720×576	600	10	Normal	Red buildings covered in smoke
Fire30	800×600	1920	15	Normal	A lab with a red front wall, where a person moves, holding a red ball
Fire31	800×600	1485	15	Normal	A lab with red tables, and a person moving with a red bag and a ball

TABLE II
COMPARISON OF VARIOUS FIRE DETECTION METHODS FOR DATASET I

Technique	False Positives	False Negatives	Accuracy
Proposed after FT	8.87%	2.12%	94.50%
Proposed before FT	9.99%	10.39%	89.8%
AlexNet after FT	9.07%	2.13%	94.39%
AlexNet before FT	9.22%	10.65%	90.06%
Foggia et al. [22]	11.67%	0%	93.55%
De Lascio et al. [25]	13.33%	0%	92.86%
Habibuglu et al. [23]	5.88%	14.29%	90.32%
Rafiee et al. (RGB) [21]	41.18%	7.14%	74.20%
Rafiee et al. (YUV) [21]	17.65%	7.14%	87.10%
Celik et al. [19]	29.41%	0%	83.87%
Chen et al. [9]	11.76%	14.29%	87.10%

Organized by

Royal College of Engineering and Technology, Thrissur, Kerala, India

the fire using Algorithm 2. These localization results are compared with those of several state-of-the-art methods, such as Chen et al. [3], Çelik and Demirel [8], Chino et al. (BoWFire) [11], Rudz et al. [12], and Rossi et al. [13].

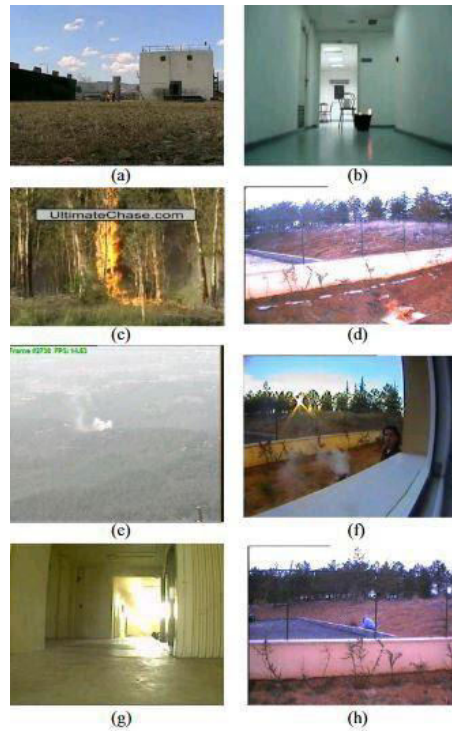


Figure 17. Set of representative images from Dataset1. The top four images are taken from videos of fires, while the remaining four images are from nonfire videos.

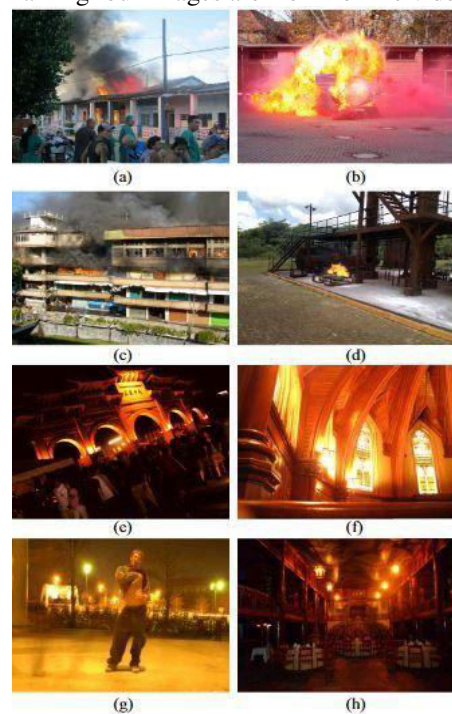


Figure 18. Representative images from Dataset2. The top four images include, fires, while the remaining four images represent fire- like normal images

We report three different results for our CNNFire based on the threshold T of the binarization process in Algorithm 2. Our approach maintains a better balance between the true positive rate and false positive rate, making it more suitable for fire localization in surveillance systems.

The results of BoWFire, color classification, Celik and Rudz are almost the same. Rossi gives the worst results in this case, and Chen is better than Rossi. The results from CNNFire are similar to the ground truth. Although BoWFire has no false positives for this case, it misses some fire regions, as is evident from its result. Color classification and Celik detect the fire regions correctly, but give larger regions as false positives. Hen fails to detect the fire regions of the ground truth image Rossi does not detect fire regions at all for this case. The false positive rate of Rudz is similar to our CNNFire, but the fire pixels detected by this approach are scarce. Although our method gives more false positives than the BoWFire method, it correctly detects the fire regions which are more similar to the ground truth images. In addition to fire detection and localization, our system can determine the severity of the detected fire and the object under observation. For this purpose, we extracted the ZOI from the input image and segmented fire regions. The ZOI image was then fed forward to the SqueezeNet model, which was trained on the ImageNet dataset with 1000 classes. The label assigned by the SqueezeNet model to the ZOI image is then combined with the severity of the fire for reporting to the fire brigade. A set of sample cases from this experiment is given in Fig.19.

VII. CONCLUSION

Fire is one of the dangerous events which can result in great losses if it is not controlled on time. This necessitates the importance of developing early fire detection systems. Recent advances in processing capabilities of smart devices have shown promising results in surveillance systems for identification of different abnormal events i.e., fire, accidents, and other emergencies., CCTV cameras are able to perform different types of processing such as object and motion detection and tracking. Considering these processing capabilities, it is possible to detect fire at its early stage during surveillance, which can be helpful to disaster management systems, avoiding huge ecological and economic losses, as well as saving a large number of human lives. Inspired by the great potential of CNNs, we propose a lightweight CNN based on the SqueezeNet architecture for fire detection in CCTV surveillance networks. This paper mainly focuses on the detection of fire and its localization, with comparatively little emphasis on understanding the objects and scenes under observation. Our approach can both localize fire and identify the object under surveillance. Furthermore, our proposed system

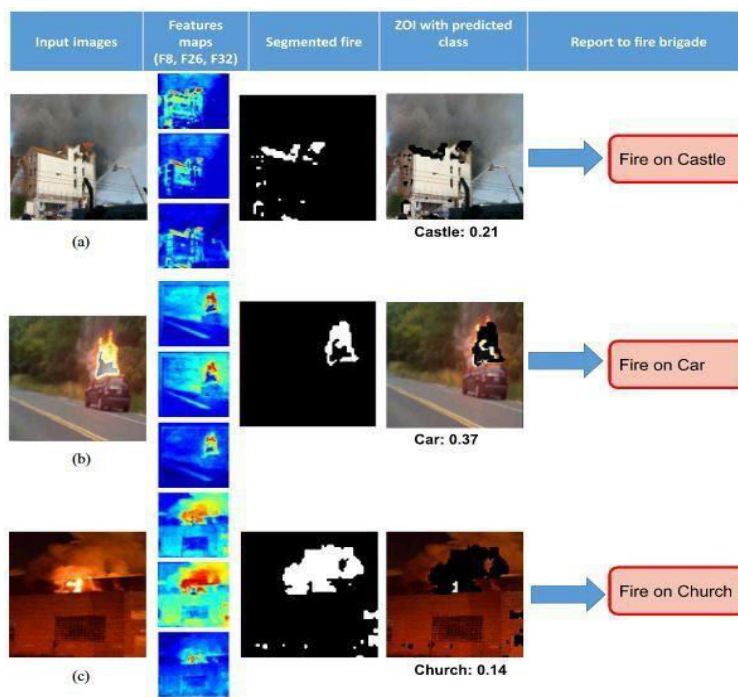


Figure 19. Sample outputs from our overall system: the first column shows input images with labels predicted by our CNN model and their probabilities, with the highest probability taken as the final class label; the second column shows

three feature maps (F8, F26, and F32) selected by Algorithm 1; the third column highlights the results for each image using Algorithm 2; the fourth column shows the severity of the fire and ZOI images with a label assigned by the SqueezeNet model; and the final column shows the alert that should be sent to emergency services, such as the fire brigade. (a) Fire: 98.76%, normal: 1.24%. (b) Fire: 98.8%, normal: 1.2%. (c) Fire: 99.53%, normal: 0.47%.

balances the accuracy of fire detection and the size of the model using fine-tuning and the SqueezeNet architecture, respectively. We conduct experiments using two bench- mark datasets and verify the feasibility of the proposed system for deployment in real CCTV networks. In view of the CNN model's reasonable accuracy for fire detection and localization, its size, and the rate of false alarms, the system can be helpful to disaster management teams in controlling fire disasters in a timely manner, thus avoid- ing huge losses. This paper mainly focuses on the detec- tion of fire and its localization, with comparatively little emphasis on understanding the objects and scenes under observation.

REFERENCES

1. B. C. KO, K.-H. Cheong, and J.-Y. Nam, "Fire detec- tion based on vision sensor and support vector ma- chines," *Fire Safety J.*, vol. 44, no. 3, pp. 322–329, 2009.
2. K. Muhammad, J. Ahmad, and S. W. Baik, "Early fire detection using convolutional neural networks dur- ing surveillance for effective disaster management," *Neurocomputing*, vol. 288, pp. 30–42, May 2018.
3. T.-H. Chen, P.-H. Wu, and Y.-C. Chiou, "An early fire-detection method based on image processing," in *Proc. Int. Conf. Image Process. (ICIP)*, 2004, pp. 1707–1710.
4. C.-B. Liu and N. Ahuja, "Vision based fire de- tection," in *Proc. 17th Int. Conf. Pattern Recognition. (ICPR)*, Cambridge, U.K., 2004, pp. 134–137.
5. G. Marbach, M. Loeffe, and T. Brupbacher, "An image processing technique for fire detection in video images," *Fire Safety J.*, vol. 41, no. 4, pp. 285–289, 2006.
6. B. U. Töreyn, Y. Dedeoğlu, U. Güdükbay, and A. E. Çetin, "Computer vision based method for real-time Fire and flame detection," *Pattern Recognition. Lett.* vol. 27, no. 1, pp. 49–58, 2006.
7. D. Han and B. Lee, "Development of early tunnel fire detection algorithm using the image processing," in *Proc. Int. Symp. Vis. Comput.*, 2006, pp. 39–48.
8. T. Çelik and H. Demirel, "Fire detection in video sequences using a generic color model," *Fire Safety J.*, vol. 44, no. 2, pp. 147–158, 2009.
9. P. V. K. Borges and E. Izquierdo, "A probabilistic approach for visionbased fire detection in videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 5, pp. 721–731, May 2010.
10. F. N. Iandola et al., "SqueezeNet: AlexNet- level accuracy with 50x fewer parameters and <1MB model size," *arXiv preprint arXiv: 1602.07360*, 2016. [Online]. Avail- able: <https://arxiv.org/pdf/1602.07360.pdf>
11. D. Y. T. Chino, L. P. S. Avalhais, J. F. Rodrigues, and
12. J. M. Traina, "BoWFire: Detection of fire in still images by integrating pixel color and texture analy- sis," in *Proc. 28th SIBGRAPI Conf. Graph. Patterns Images*, 2015, pp. 95–102.
13. S. Rudz, K. Chetehouna, A. Hafiane, H. Laurent, and
14. O. Séro-Guillaume, "Investigation of a novel image segmentation method dedicated to forest fire applica- tions," *Meas. Sci. Technol.*, vol. 24, no. 7, 2013, Art. no. 075403.
15. L. Rossi, M. Akhloufi, and Y. Tison, "On the use of stereovision to develop a novel instrumentation sys- tem to extract geometric fire fronts characteristics," *Fire Safety J.*, vol. 46, nos. 1–2, pp. 9–20, 2011.
16. P. Foggia , A. Saggese , M. Vento , Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion, *IEEE Trans. Circuits Syst. Video Technol.* 25 (2015) 1545–1556 .
17. D.Y. Chino , L.P. Avalhais , J.F. Rodrigues , A.J. Traina , BoWFire: detection of fire in still images by integrating pixel color and texture analysis, in: Pro- ceedings of the 28th 2015 Conference on Graphics, Patterns and Images, SIBGRAPI , 2015, pp. 95–102
18. M. Sun, Z. Song, X. Jiang, J. Pan, Y. Pang, Learning pooling for convolutional neural network, *Neurocomputing* 224 (2017) 96–104.
19. A.O. Bicen, V.C. Gungor, O.B. Akan, Delay- sensitive and multimedia communication in cogni- tive radio sensor networks, *Ad Hoc Netw.* 10 (2012) 816–830
20. D.Y. Chino , L.P. Avalhais , J.F. Rodrigues , A.J. Traina , BoWFire: detection of fire in still images by integrating pixel color and texture analysis, in: Pro- ceedings of the 28th 2015 Conference on Graphics,
21. *Patterns and Images, SIBGRAPI*, 2015, pp. 95–102
22. K. Muhammad, M. Sajjad, M.Y. Lee, S.W. Baik, Efficient visual attention driven framework for key frames extraction from hysteroscopy videos, *Biomed. Signal Process. Control* 33 (2017) 161–168[53]



National Conference on Emerging Technologies [NCET' 2021]

Organized by

Royal College of Engineering and Technology, Thrissur, Kerala, India

23. K. Muhammad , J. Ahmad , M. Sajjad , S.W. Baik, Visual saliency models for summarization of diagnostic hysteroscopy videos in healthcare systems, Springer- plus 5 (2016) 1495.
24. R. Di Lascio , A. Greco , A. Saggese , M. Vento
25. Improving fire detection reliability by a combination of videoanalytics, in: Proceedings of International Conference on Image Analysis and Recognition, 2014, pp. 477–484.
26. Y.H. Habibo vglu , O. Günay , A.E. Çetin , Covariance matrix-based fire and flame detection method in video, Mach. Vis. Appl. 23 (2012) 1103–1113 .
27. T. Toulouse, L. Rossi, M. Akhloufi, T. Celik, and X. Maldague, “Benchmarking of wildland fire colour segmentation algorithms,” IET Image Process., vol. 9, no. 12, pp. 1064–1072, Dec. 2015.
28. C.-B. Liu and N. Ahuja, “Vision based fire detection,” in Proc. 17th Int.Conf. Pattern Recognit. (ICPR), Cambridge, U.K., 2004, pp. 134–137.
29. T. Celik, H. Demirel, H. Ozkaramanli, and M. Uyguroglu, “Fire detection using statistical color model in video sequences,” J. Vis. Commun. Image Represent. vol. 18, no. 2, pp. 176–185, 2007.



**Impact Factor:
7.569**



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details