International Journal on Emerging Technologies **10**(2): 450-454(2019)

ISSN No. (Print) : 0975-8364
ISSN No. (Online) : 2249-3255

Convolution Neural Network based Fire Detection in Surveillance Videos

***A. Umamakeswari*[1], *Venkatesh Prasad*[2], *Vivek Pothina*[2] *and S.M. Narendran*[3]**
[1]*Dean, School of Computing, SASTRA University, Tamil Nadu, India.*
[2]*B-Tech, Department of Computer Science, SASTRA University, Tamil Nadu, India.*
[3]*Research Scholar, School of Computing, SASTRA University, Tamil Nadu, India.*

(Corresponding author: A. Umamakeswari)
(Received 01 June 2019, Revised 02 August 2019 Accepted 19 August 2019)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Surveillance has many applications in Intrusion detection, theft detection, and event monitoring. During surveillance, many abnormal events can occur such as fire, accidents, and disasters. Among such abnormal events, fire is one of the most commonly happening events. This necessity the existence of effective fire alarming systems for surveillance. The existing flame detection systems rely on visible light from surveillance cameras. The existing methods can be categorized into three categories: pixel-level, blob-level, and patch-level methods. But overall these methods don't work on small fires and color-based methods are sensitive to fire-colored objects, brightness, and shadows. As a result, the number of false warnings produced by these methods is high. In this paper, a Convolutional Neural Network based on a pre-trained GoogLeNet model which won the ImageNet Challenge in 2014, this model is trained on the mivia dataset of Fire and Not-Fire videos. The model is tweaked until maximum accuracy is achieved. This paper is focused on a solution to provide fire detection with existing infrastructure of CCTVs and surveillance systems.

Keywords: Internet of Things, Deep Learning, CCTV video Analysis, Convolutional Neural Network, Fire Detection, Transfer Learning, Real-World Application, Image Classification, Pre-Trained Model

I. INTRODUCTION

Fires are chemical processes in which oxidation takes place and releases smoke, light, and heat. Currently, the majority of domestic automatic fire alarm systems use a single passive sensor alarm system, which has some unavoidable problems. For example, some devices using photosensitive detectors are affected by sunlight and lighting [2]. Simple fire detection systems using threshold algorithms are not robust as they can be affected by various gases. These systems might work effectively in simple situations, the false alarm rate based on simple algorithms will be great, so alarm systems will then result in many leak-checks or false fire alarms. Thus, traditional fire detection systems are unable to meet the needs of real fire alarms [3].

Fire disasters mainly occur due to human error or the failure of a system, causing economic as well as ecological damage along with endangering human lives [4]. According to Guha-Sapir *et al.,* [5], wildfire disasters alone in the year 2015 resulted in 494,000 victims and caused damage worth US$ 3.1 billion. Each year, an area of vegetation of 10,000 km^2 is affected by fire disasters in Europe. Considering all these, the detection of fire should be done effectively so that damages from disasters can be reduced. After all, fire is one of the common and most affecting elements of almost all calamities like earthquakes and man-made disasters which will involve explosions.

The pixel-level methods are fast due to the usage of pixel-wise features such as colors and flickers, however, their performance is not attractive as such methods can be easily biased. Blob level methods consider blob-level candidates for feature extraction. The disadvantage is the difficulty in training due to numerous shapes of fire blobs. Patch-level algorithms are developed to improve the performance; however, such methods result in many outliers, affecting their accuracy. Other methods include using color and motion features like using both RGB and HSI color spaces and checking features frame by frame, another method investigates temporal and spatial wavelet analysis.

Deep Learning models are getting more robust and some state-of-the-art models are more accurate than humans. Especially, Convolutional Neural Networks are producing very good results with computer vision use cases as feature extraction using CNNs is providing good results. This paper discusses one such use case, implementing CNN model to detect Fire in Surveillance videos.

II. PREVIOUS WORKS

So far, from the available literature fire or flame detection using visible light from the camera, which can be categorized into three categories: pixel-level, blob-level, and patch-level methods. The pixel-level methods [6], are fast because of the usage of features like colors and flickers which are pixel-wise, however, their performance is not good and moreover, these methods can easily be biased [7]. Compared to pixel-level methods, blob-level flame detection methods [8] show better performance as such methods consider blob-level candidates for feature extraction to detect the flame. The difficult part of these methods lies in the training stage while training blob-level classifiers. The training is difficult due to numerous shapes of fire blobs. Patch-level algorithms [9] are an improvement over the performance of the previous two categories of flame detection algorithms, but such methods result in many outliers, affecting their accuracy.

To improve the accuracy, color and motion features have been explored for flame detection. Chen *et al.* [7] investigated the dynamic behavior and irregularity of flames in both RGB and HSI color spaces for fire detection. Their method failed to make a difference of real fire from fire-like moving outliers and objects because their method considers the frame difference during prediction. Besides RGB and HSI color models, Marbach *et al.* [10] explored the YUV color model in combination with motion features for the prediction of fire and non-fire pixels. A similar method is proposed by Töreyin *et al.* [8] 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 [11] by comparing the video frames and their color features for flame detection in tunnels. Continuing the investigation of color models, Celik and Demirel [12] used YCbCr with specific rules of separating chrominance components from luminance. The method has the potential to detect flames with good accuracy but at a 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 [13].

In the continuation of Borges and Izquierdo work, multi-resolution 2D wavelets combined with energy and shape are explored by Rafiee *et al.,* in an attempt to reduce false warnings, however, the false fire alarms still remained significant due to the movement of rigid body objects in the scene [14]. An improved version of this approach is presented by using YUC instead of the RGB color model, providing better results than the previous. To further improve the accuracy, Foggia *et al.* 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 alarm rate is still high and can be further reduced [15].

III. MODEL ARCHITECTURE

The proposed architecture is to collect data from video surveillance video or CCTV and process them in real-time. A pre-trained CNN model is used for classification at real-time, classifying the video frame by frame into the fire and no fire classes. This pre-trained CNN model can be run in a remote server to which aggregated data comes from different video surveillance systems and the CNN output real-time predictions on the streaming data. The data storage is used to fill data as the video keeps streaming. As the data increases, soon the model will have more data to train and test. By this architecture, the dataset will be getting rich and the model will be getting better at predicting with more accuracy. This architecture to detect fires is considered to be cost-effective as there won't be a big change in the existing infrastructure, data from existing CCTVs and various surveillance systems are used to detect fire. The relative cost of this architecture to coming up with new infrastructure for fire detection makes a huge difference. Fig. 1 shows the architecture of the model.

A. Convolutional Neural Network

Convolutional Neural Network has become really popular for image classification since the LeNet [16] performed well on the MNIST Data (Handwritten Digits Dataset) and achieved an accuracy which is more than humans. Since then, CNN has performed at a state-of-the-art level at image classification tasks.

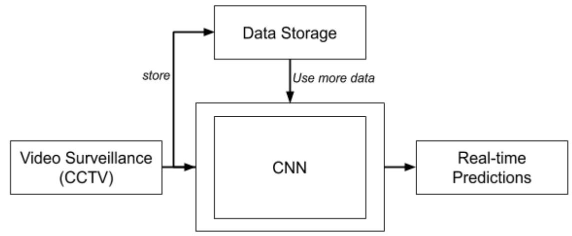

Fig. 1. Model Architecture.

A typical CNN consists of different types of processing layers including convolution, pooling, and fully connected. These layers are arranged in such a way that the output of one layer becomes the input of the next layer. At each convolution layer, a number of kernels are applied to the input data to generate feature maps. Pooling layers select maximum activations within small neighborhoods of these features maps to reduce and introduce translation invariance. Fully connected layers followed by stacks of convolutional and pooling layers model high-level abstractions in the data and serve as high-level representations of the input. The weights of all the convolutional kernels and neurons in the fully connected layers are learned during the training process and correspond to essential characteristics of the training data, useful for performing the intended classification.

B. Transfer Learning

Compared to Traditional Machine Learning methods, Deep Learning has a very strong dependence on Massive training data as it needs huge amounts of data to understand the latent patterns present in data. Transfer Learning solves this issue in one way. In this way, models that are trained on huge data is taken, often this data in the same scope of the current use case. The last layer of the neural network is removed and trained on the new and smaller data that is present for the current use case. This method is very useful as many CNN models have been trained in various Image and Object Recognition competitions, these pre-trained models can be used.

C. GoogleNet

The winner of the ILSVRC 2014 competition was GoogLeNet [1] (also known as Inception V1) from Google. It achieved an error rate of 6.67%. This was very close to human-level performance which the organizers of the challenge were now forced to evaluate. As it turns out, this was actually rather hard to do and required some human training in order to beat GoogLeNets accuracy. The network used a CNN inspired by LeNet [16] but implemented a novel element that is dubbed an inception module. It used batch normalization, image distortions, and RMSprop. This module is based on several very small convolutions in order to drastically reduce the number of parameters.

Their architecture consisted of a 22 layer deep CNN but reduced the number of parameters from 60 million to 4 million. There are many advantages of using GoogleNet which has better classification accuracy compared to other models like LeNet, AlexNet and also, it's a small-sized model which is also suitable for implementation on

FPGAs and other hardware architectures having memory constraints.

For transfer learning, last output layer of this model is removed. A new output layer is added and the model is trained. With existing pre-trained connections, CNN needs less dataset to achieve better accuracy.

Dataset is collected by Foggia *et al*. [15], containing 31 videos that cover different environments.

Fig. 2. mivia Dataset Examples.

The dataset is composed of two main parts: the first 14 videos characterized by the presence of fire and the last 17 videos that do not contain fires; in particular, this second part is characterized by objects or situations, which can be wrongly classified as containing fire: a scene containing red objects may be misclassified by color-based approaches, while a mountain with smoke, fog, or clouds may be misclassified by motion-based approaches.

These videos have been converted into a dataset of over 47,000 images by splitting the video frame by frame. The exact number of images with fire is 11039 and the number of images without fire is 36794. For training, 60% of the dataset has been used, 20% for testing and the remaining 20% for validation.

The dataset has been made challenging for both color-based and motion-based fire detection methods by capturing videos of fire-like objects and mountains with smoke and clouds.

IV. METHODOLGY

For the implementation part of the paper, Python has been used extensively. With its relatively easy learning curve and many open-source Machine Learning Frameworks, Python was used in experimenting and prototyping. Out of the many Machine Learning Frameworks, Keras was chosen due to the availability of many pre-trained models. Google Colab, is a research tool for machine learning education and research developed by Google. It's a Jupyter notebook environment that comes with pre-installed Machine Learning tools and libraries. And it is also free to use for research purposes. The dataset has been loaded into Google Colab, pre-processed, split into training, validation and testing set, then used for training pre-trained models and then tested. As the project is based mostly on Python, Matplotlib library is used for data visualization.

A. Preprocessing

The mivia dataset is downloaded and loaded into Google Colab. The dataset comprises of 31 videos with 14 fire videos and 17 non-fire videos. Using OpenCV library, the video frames has been split into images and stored in respective labeled directories. The data in these labels are further split into training, validation and testing datasets. One of the standard ratios was used, 60% for training, 20% for validation and 20% for testing.

B. Training and Testing

VGG16 Model of Keras is used as the pre-trained model. It is similar to Inception V1 or GoogLeNet Model which won the ImageNet Challenge 2014. The last layer is removed and a standard output layer of 2 perceptrons is added. The model is trained on the training dataset batch-wise.

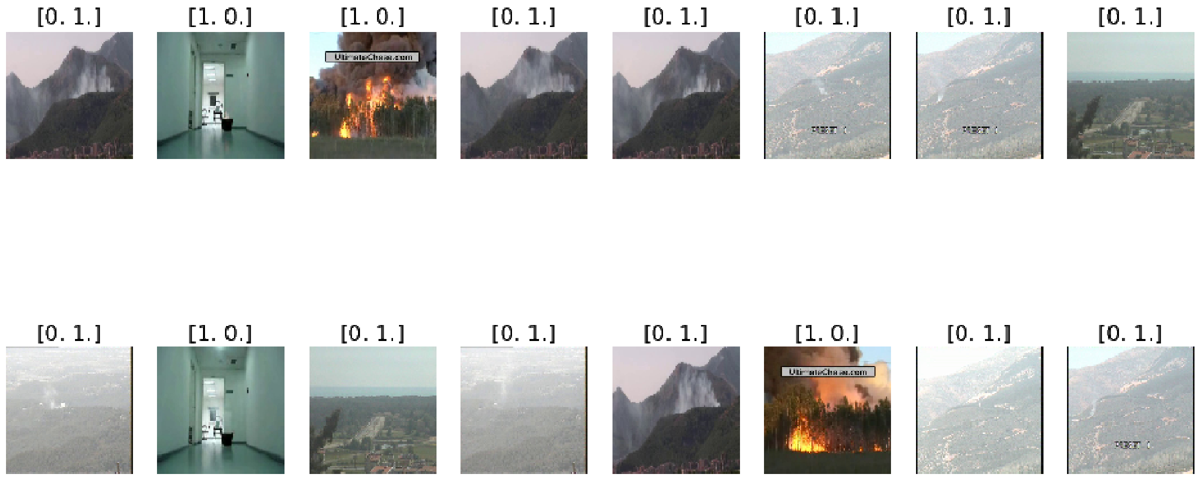

Fig. 3. Predictions made on test data.

After many trial and error, the model has set up with Adam optimizer, a learning rate of 0.001. Categorical Cross Entropy is used as the loss function. After training, the model is tested on the test dataset. Fig. 3 shows some of the predictions (output) made by the model on the test data.

IV. RESULTS AND OBSERVATIONS

The model was developed in Keras and run on a Google Colab GPU for training and testing. The validation accuracy obtained on this dataset for the proposed model is 96% and a testing accuracy of 92.7%, which can be seen in Fig. 4.

```
t = model.evaluate_generator(test_batches, 6)
print("The Accuracy is ", t[1])

The Accuracy is  0.9270833333333334
```

Fig. 4. Accuracy on test dataset.

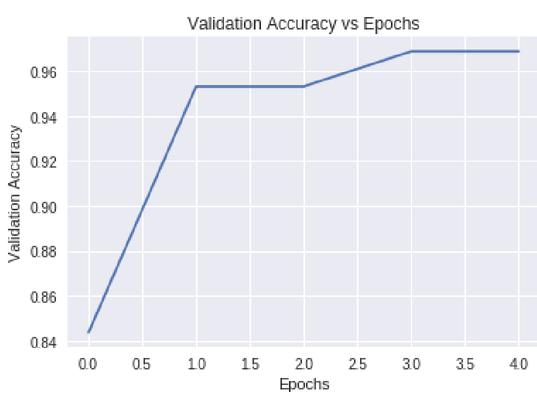

Fig. 5. Validation Accuracy plot of the model.

Fig. 5 depicts the validation accuracy vs epochs plot for the developed model. And figure 5 depicts the loss vs epochs for the model during the learning phase of the model.

The number of epochs was chosen to be 30, as a conclusion was made from the validation vs epochs graph (Fig. 5) that the model might be overfitting on the relatively smaller dataset after 30 epochs.

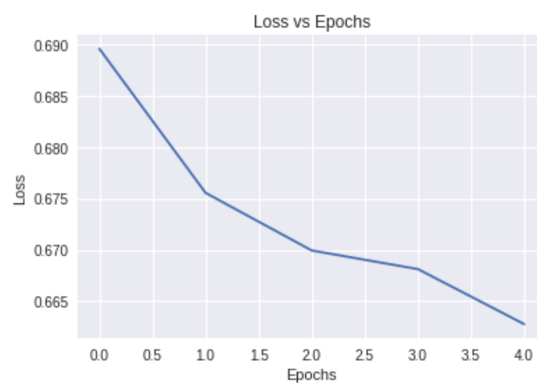

Fig. 6. Loss Function of the model.

IV. CONCLUSION

This method of processing capabilities of smart devices has shown promising results in surveillance systems for the identification of fire accidents. Fire is one of the most 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. Therefore, in this paper, we used a cost-effective fire detection CNN architecture for surveillance videos. The model is inspired by GoogleNet architecture and is fine-tuned with a special focus on computational complexity and detection accuracy. It is proved that the proposed architecture dominates the existing hand-crafted features based fire detection methods which rely on basic image processing techniques.

V. FUTURE SCOPE

It is a need to tackle the forest fire with some advanced technologies like artificial intelligence. CNN may be clubbed with technologies like LSTM to make the knowledge available for the rest of the future with the past historical data.

REFERENCES

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

[2]. Cheng, C., Sun, F., & Zhou, X. (2011). One fire detection method using neural networks. *Tsinghua science and technology*, **16**(1), 31-35.

[3]. Zang, X., Wang, Y., Luo, Y., & Song, S. (2000). Study on fire alert detecting system on fuzzy-neural network. *Journal of China Coal Society*, **12**(3), 283-286.

[4]. Chen, T.H., Wu, P.H., & Chiou, Y.C. (2004). An early fire-detection method based on image processing. In *2004 International Conference on Image Processing, 2004. ICIP'04.* (Vol. **3**, pp. 1707-1710). IEEE.

[5]. Guha-Sapir, D., Vos, F., Below, R., Penserre, S. Annual disaster statistical review 2015: the numbers and trends, 2015.

[6]. Liu, C.B. and Ahuja, N. (2004). Vision based fire detection. *In Proc. 17th Int. Conf. Pattern Recognit. (ICPR),* Aug. 2004, pp. 134–137.

[7]. Chen, T.H., Wu, P.H. and Chiou, Y.C. (2004). An early fire-detection method based on image processing. *Proc. Int. Conf. Image Process. (ICIP),* Oct. 2004, pp. 1707–1710.

[8]. Töreyin, B. U., Dedeoğlu, Y., Güdükbay, U., & Cetin, A. E. (2006). Computer vision based method for real-time fire and flame detection. *Pattern recognition letters*, **27**(1), 49-58.

[9]. Choi, J., & Choi, J.Y. (2015). Patch-based fire detection with online outlier learning. In *2015 12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)* (pp. 1-6). IEEE.

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

[11]. Han, D., & Lee, B. (2006). Development of early tunnel fire detection algorithm using the image processing. In *International Symposium on Visual Computing* (pp. 39-48). Springer, Berlin, Heidelberg.

[12]. Celik, T., & Demirel, H. (2009). Fire detection in video sequences using a generic color model. *Fire Safety Journal*, *44*(2), 147-158.

[13]. Borges, P. V. K., & Izquierdo, E. (2010). A probabilistic approach for vision-based fire detection in videos. *IEEE transactions on circuits and systems for video technology*, *20*(5), 721-731.

[14]. Rafiee, A., Dianat, R., Jamshidi, M., Tavakoli, R., & Abbaspour, S. (2011). Fire and smoke detection using wavelet analysis and disorder characteristics. In *2011 3rd International Conference on Computer Research and Development* (Vol. **3**, pp. 262-265). IEEE.

[15]. Foggia, P., Saggese, A., & Vento, M. (2015). Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion. *IEEE TRANSACTIONS on circuits and systems for video technology*, **25**(9), 1545-1556.

[16]. LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E., & Jackel, L.D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems* (pp. 396-404).

[17]. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

How to cite this article: Umamakeswari, A., Prasad, Venkatesh, Pothina, Vivek and Narendran, S.M. (2019). Convolution Neural Network based Fire Detection in Surveillance Videos. *International Journal of Emerging Technologies*, **10**(2): 450–454.