AUTOMATED FIRE DETECTION AND SURVEILLANCE SYSTEM

K. Ramya Laxmi
Assistant Professor, CSE, Sreyas institute of engineering and Technology, Telangana, India
ramya.kunta@sreyas.ac.in

A. Sreeja
CSE, Sreyas institute of engineering and Technology, Telangana, India
sreejaannapureddy2011@gmail.com

E. Revanth
CSE, Sreyas institute of engineering and Technology, Telangana, India
grevanth571@gmail.com

Manasa Gourishetty
CSE, Sreyas institute of engineering and Technology, Telangana, India
manasagourishetty003@gmail.com

M. Rushikesh
CSE, Sreyas institute of engineering and Technology, Telangana, India
rushigoud2000@gmail.com

ABSTRACT:
The major goal of this project is to build a fire detection and surveillance system that is automatic. During surveillance, Convolutional Neural Networks will be employed to detect the fire (CNNs). Such methods, on the other hand, typically need greater processing time and memory, limiting their use in surveillance networks. We propose a low-cost fire detection CNN architecture for surveillance films in this research. This is mostly concerned with computational complexity and detection precision. The model is fine-tuned to balance efficiency and accuracy, taking into account the nature of the target problem and fire data. This system takes an image or video file as input and detects fire and fire percentages that are precise enough to prevent fire mishaps and save human lives.

KEYWORDS:
Automatic fire detection, Convolutional Neural Networks, Surveillance system

1. INTRODUCTION

Smarter surveillance has resulted from smarter embedded processing capabilities of smart devices, resulting in a variety of helpful applications in various fields such as e-health, autonomous driving, and event monitoring. During surveillance, different abnormal events can occur such as fire, accident, disaster, medical emergency, flood about which early information is important. This can greatly minimize changes of big disasters and can control an abnormal event on time with comparatively minimum possible loss

The delayed escape for disabled people as the traditional fire alarming system need strong fires or close proximity, failing to generate an alarm on time for such people. This involves the installation of effective fire alarming and surveillance systems. The fire alarming systems can be developed based on vision sensors, considering its affordable cost and installation. The best way to improve fire detection is by detecting fire using cameras

The advent of Artificial Intelligence (AI) changed the world in every way. Machine learning (ML), a subset of AI helps the human to find solutions for highly complex problems and also plays a vital role in detecting the fire. The idea is to determine the spread of fire in villages and sub-urban areas, where fire extinguisher might not be readily available. We want to build a machine learning model that could predict fire. Our approach predicts fire using dataset having parameters and inputs.

Our main motive is to use Machine Learning Algorithms in detection of fire. And to increase the accuracy of prediction. Our model would be beneficial for the prevention of loss by assisting them take the appropriate action in terms of assuring that enough resources are available.

2. EXISTING SYSTEM

The most often used fire detection method, according to current literature, is flame detection utilizing a visible light camera. which is divided into three categories: pixel-level, blob-level, and patch-level methods. Due to the use of pixel-wise properties such as colors and flickers, pixel- level approaches are quick., However, their performance is unappealing since such methodologies are easily skewed. Blob-level flame detection approaches outperform pixellevel methods because they evaluate blob-level candidates for feature extraction while detecting flame. Due to the diverse shapes of fire blobs, such approaches have a tough time training their classifiers. Patch-level techniques are designed to increase the accuracy of the preceding two types of flame detection algorithms; nevertheless, such systems produce a large number of outliers, reducing their accuracy.

LIMITATIONS OF EXISTING SYSTEM:

- The fire is discovered just after smoke is detected.
- Even if there is a fire, the smoke may not appear for a long time after the surrounds have been burned.
- The smoke detectors take a long time to detect the smoke.
- Materials in the immediate vicinity will be burned until the next preventive action is taken.

3. PROPOSED SYSTEM:

Python is being used to build the proposed system. The system is made capable of detecting, evaluating and measuring the outbreak of fire using some predefined parameters which are taken in count.

Almost all systems predict fire using dataset having parameters and inputs. None of the systems predicts fire based on risk factors. Our model would be beneficial for all by assisting them take the appropriate action in terms of assuring that enough resources.

- Minimize financial burden on fire stations by providing them first-hand information about area affected by fire.
- The feature vectors in our model are general enough to be adapted with a slight change to detect fire.
- Detecting fire using various Risk Factors can result into a more accurate outcome and can everyone.

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

To deal with these challenges, we analyzed and investigated Deep Learning architectures for early flame detection in depth. We studied a number of CNNs to increase the flame detection accuracy

and reduce the incidence of false alerts, motivated by recent improvements in embedded processing capabilities and the promise of deep features.

3.1. PROPOSED ARCHITECTURE:

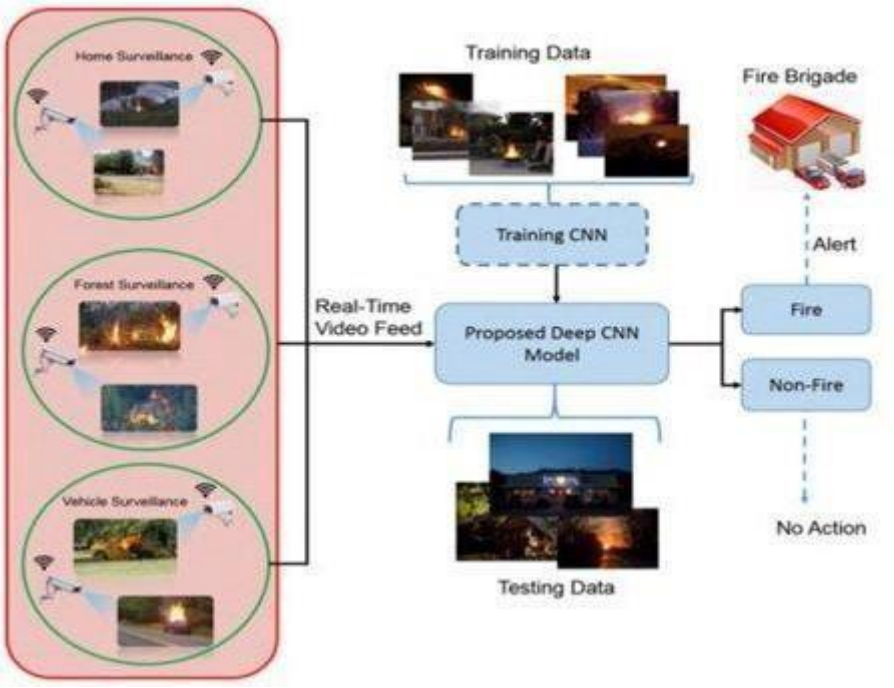

Fig.1. System Architecture

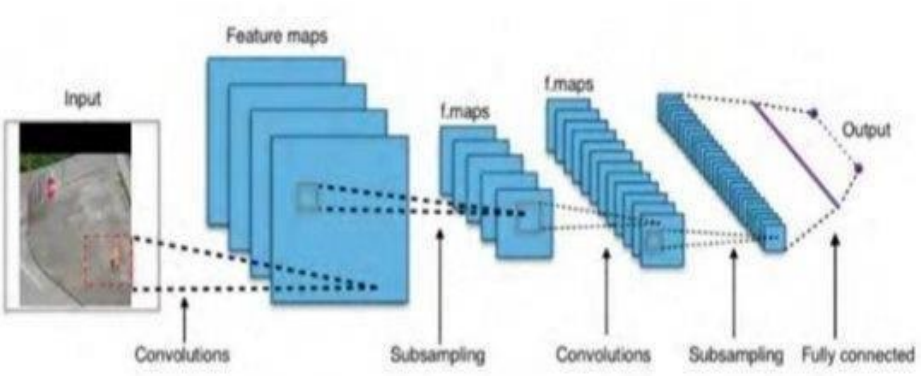

Fig.2. Technical Architecture

A simple ConvNet is a series of layers, each of which uses a differentiable function to translate one volume of activations to another. Convolutional Layer, Pooling Layer, and Fully Connected Layer are the three types of layers we employ to develop ConvNet topologies. These layers will be stacked to construct a complete ConvNet architecture. The architecture of a simple ConvNet classification may be [INPUT - CONV(convolutional) - ReLU (rectified linear unit)- POOL (pooling)- FC(fully connected)].

More specifically:

•INPUT [32x32x3] will save the picture's raw pixel values, in this example an image with a width of 32 pixels, a height of 32 pixels, and three colour channels (R,G,B).

•The output of neurons connected to particular regions in the input will be computed by the CONV layer, which will compute a dot product between their weights and they are connected to a small region in the input volume. If we chose to employ 12 filters, this may result in a volume of [32x32x12].

•The elementwise activation function, such as (max(0,x)) thresholding at zero, will be applied by the RELU layer. The volume's size remains unchanged as a result ([32x32x12]).

- The POOL layer will downscale along the spatial dimensions (width and height), resulting in a volume of [16x16x12].
- The class scores will be computed by the FC (fully-connected) layer, resulting in a volume of size [1x1x10], where each of the ten values corresponds to a class score, such as among the CIFAR-10 types. Each neuron in this layer will be connected to all the numbers in the preceding volume, just like in regular Neural Networks.

ConvNets do this by layering the original image from the initial pixel values to the final class scores. It's worth noting that certain layers have parameters while others don't. The CONV/FC layers, in particular, perform transformations based on both the input volume activations and the parameters (the weights and biases of the neurons).. The RELU/POOL layers, on the other hand, will implement a fixed function. Gradient descent will be used to train the parameters in the CONV/FC layers so that the class scores computed by the ConvNet are consistent with the labels in the training set for each image.

3.2. IMPLEMENTATION:

All of the activities that occur during the transition from the old to the new system are referred to as implementation. The old system consists of manual operations, which is operated in a very difficult manner from the proposed system. A proper implementation is essential to provide a reliable system to meet the requirements of the organization.

Python:

- The Python programming language is an Open Source, cross-platform, high level, dynamic, interpreted language.
- Python is a very flexible language. It's commonly utilized for a variety of reasons.

Matplotlib Library:

- Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.
- matplotlib.pyplot is a collection of methods that make matplotlib act like MATLAB. Each pyplot function alters a figure in some way, such as constructing a figure, a plotting area in a figure, charting specified lines in a plotting area, and so on.

Pillow Python Imaging Library (Fork) :

- The Python Imaging Library extends your Python interpreter's image processing capabilities.
- This library can read and write a wide range of file formats, has a fast internal representation, and can conduct image processing.
- The foundation of the picture library is designed to allow easy access to data preserved in a few basic pixel formats. It should be a suitable starting point for any image processing software.

TensorFlow:

Google designed and distributed TensorFlow, a Python library for fast numerical computing. It's a foundation library for developing Deep Learning models, either directly or through wrapper libraries developed on top of TensorFlow to make the process easier.

OpenCV

OpenCV is an open-source image processing, computer vision, and machine learning library. Python, C++, Java, and other programming languages are supported by OpenCV. It can recognize products, persons, and even human handwriting in photographs and videos. It becomes an extremely efficient numerical operations library when paired with other libraries, such as NumPy. Because all of NumPy's operations may be combined with OpenCV, your arsenal of weaponry grows.

Importing necessary Python Libraries:

- We imported NumPy, Pandas for data frames.
- We also imported visualization libraries such as matplotlib, matplotlib.pyplot.
- we import other necessary libraries as TensorFlow, pillow and OpenCV **Import Dataset:**
- Import the dataset from the computer using function *“pd.read_csv”*.
- The "read csv" method imports a csv dataset and stores it in an Environment variable.

Example: *Obj_detection_tutorial.read= Obj_detection_tutorial.read_csv(C:\miniproject\research\ )* **Tensor application** *import TensorFlow as tf tensor_name = key + ':0'*

if tensor_name in all_tensor_names:

tensor_dict[key] = tf.get_default_graph().get_tensor_by_name(tensor_name) **Pillow Package (loading image datasets)** *from PIL import Image*

def load_image_into_numpy_array(image):
print("shape",image.size)(return np.array(image.getdata()).reshape((im_height, im_width, .astyim_width, im_height) = image.sizepe(np.uint8)

4.RESULT:

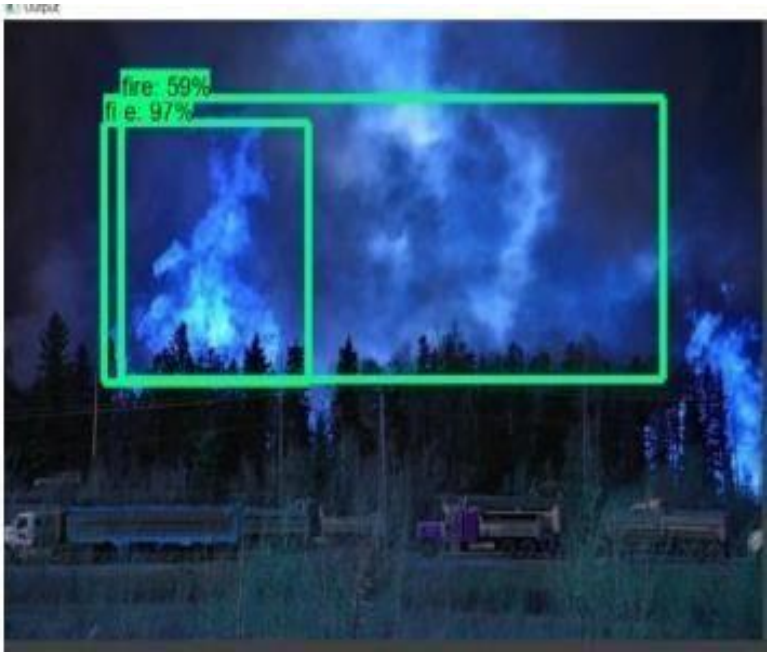

Fig.3.Result

5. REFERENCES:

1. *K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H. H. Ge Wang, and S. W. Baik, ‘‘secure surveillance framework for IoT systems using probabilistic image encryption,’’*
2. *K. Muhammad, J. Ahmad, and S. W. Baik, ‘‘Early fire detection using convolutional neural networks during surveillance for effective disaster management,’’ Neurocomputing, vol. 288, pp. 30–42, May 2018.*
3. *J. Choi and J. Y. Choi, ‘‘An integrated framework for 24-hours fire detection,’’ in Proc. Eur. Conf. Compute. Vis., 2016, pp. 463–479.*
4. *P. V. K. Borges and E. Izquierdo, ‘‘A Probabilistic approach for vision based fire detection in videos,’’ , pp. 721–731, May 2010.*
5. *A. Rafiee, R. Dianat, M. Jamshidi, R. Tavakoli, and S. Abbaspour, ‘‘Fire and smoke detection using wavelet analysis and disorder characteristics,’’ Develop. (ICCRD), Mar. 2011, pp. 262–265.*