Journal for Educators, Teachers and Trainers
The LabOSfor electronic, peer-reviewed, open-access Magazine

Journal for Educators, Teachers and Trainers
The PROFESIOLab electronic, peer-reviewed, open-access Magazine

ISSN 1989-9572

DOI: 10.47750/jett.2021.12.04.36

AI-Powered Fire Detection: Enhancing Early Warning Systems with Smart Cameras and Deep Learning

L. Ramesh1, T. Ravinder 2, K. Gopi Krishna 3

Journal for Educators, Teachers and Trainers, Vol. 12 (4)

https://jett.labosfor.com/

Date of reception: 15 Oct 2021

Date of revision: 04 Nov 2021

Date of acceptance: 05 Dec 2021

L. Ramesh1, T. Ravinder 2, K. Gopi Krishna 3 (2021). AI-Powered Fire Detection: Enhancing Early Warning Systems with Smart Cameras and Deep Learning. Journal for Educators, Teachers and Trainers, Vol. 12(4)276-292

Journal for Educators, Teachers and Trainers
The LabOSfor electronic, peer-reviewed, open-access Magazine

Journal for Educators, Teachers and Trainers
The LabOSfor electronic, peer-reviewed, open-access Magazine

Journal for Educators, Teachers and Trainers, Vol. 12 (4)
ISSN 1989 –9572

https://jett.labosfor.com/

AI-Powered Fire Detection: Enhancing Early Warning Systems with Smart Cameras and Deep Learning

L. Ramesh[1], T. Ravinder [2], K. Gopi Krishna [3]

[1]Department of Electronics and Communication Engineering

[1]Sree Dattha Institute of Engineering and Science, Sheriguda, Hyderabad, Telangana

ABSTRACT

A single passive sensor alarm system, which has some inevitable issues, is currently used by the majority of automatic fire alarm systems. Certain components in home fire alarm systems that use photosensitive detectors are impacted by illumination and sunlight. Different gasses can have an impact on smoke detectors. We frequently receive false fire alarms as a result. As a result, conventional fire detection systems cannot satisfy the requirements of an actual fire alarm. AI fire detectors are capable of detecting smoke and both small and large fire targets. Compared to sensors, it can offer better spatial awareness and visual information. AI cameras can aid in fire detection and stop wildfires from spreading. Buildings can be equipped with AI cameras to keep an eye on the interior environment. On-site incident command can be enhanced with the use of AI cameras. Heat, flame, gas, and smoke sensors are the foundation of the current fire detection system. The current system's drawbacks include excessive power consumption, expensive maintenance, and interference from the environment. In addition to overcoming the shortcomings of the current system, the suggested AI fire detection can identify smoke and flames in pictures or videos. Early fire detection skills have been transformed by fire detection systems that include AI-enabled cameras, providing a proactive approach to fire protection. By using cutting-edge computer vision techniques and deep learning algorithms to examine live video streams from security cameras, these systems are able to quickly and accurately identify patterns of smoke and fire.

Keywords: Fire Detection, Image Preprocessing, AI-Enabled cameras, Classification.

1. INTRODUCTION

Fire detection from images is a critical application of computer vision and artificial intelligence that aims to identify and alert authorities or individuals about the presence of fire or smoke in visual data. This technology is employed in various contexts, including industrial facilities, surveillance systems, and even wildfire monitoring.

The process of fire detection from images typically involves several key steps. First, image or video data is acquired through cameras or other visual sensors. Next, the data is preprocessed to enhance image quality and reduce noise, making it suitable for analysis. Feature extraction techniques are then applied to identify relevant patterns, such as flames, smoke, or heat sources. These features are used as inputs for machine learning algorithms, including convolutional neural networks (CNNs), which are particularly effective for image analysis tasks.

The trained machine learning model processes the image data and generates predictions about the presence of fire or smoke. These predictions can be binary (fire or no fire) or multi-class (e.g., fire, smoke, no fire). To ensure accuracy and reliability, the model is typically trained on a diverse dataset containing various fire scenarios, lighting conditions, and environments.

Fire detection systems can employ real-time monitoring, continuously analyzing images or video streams and triggering alarms or notifications when fire or smoke is detected. This immediate response can be crucial for timely firefighting efforts and safety measures. Additionally, integration with other systems, such as fire suppression systems or emergency services, can further enhance the effectiveness of fire detection from images.

So, fire detection from images is a vital technology that enhances safety and security across a wide range of applications. It leverages the power of computer vision and machine learning to swiftly identify potential fire hazards, enabling prompt responses that can help mitigate damage and save lives. Advances in this field continue to improve the accuracy and speed of fire detection systems, making them indispensable in fire prevention and control efforts.

2. LITERATURE SURVEY

Zhang et.al [1] The review paper analyzed 37 research articles on deep learning (DL) models for forest fire detection, which had been published between January 2018 and 2022. It delved into data types, including images and videos, data augmentation methods, and DL model architectures. Structured into five sections—classification, detection, detection and classification, segmentation, and segmentation and classification—the paper evaluated model performance using metrics like accuracy and F1-Score. Favorable outcomes emerged, with the majority of studies having achieved accuracy rates exceeding 90%. The paper recommended refining models through hyperparameter fine-tuning, integrating satellite data, employing generative data augmentation, and optimizing DL architectures. It emphasized DL's potential in crucial forest fire management.

Zhao et.al [2] In response to challenges, we introduced the Fire Segmentation-Detection Framework (FSDF), blending traditional methods with deep learning. FSDF improved flame feature detection using Hue, Saturation, and Value (HSV) and the Complete Local Binary Pattern (CLBP). We integrated YOLOv8 and Vector Quantized Variational Autoencoders (VQ-VAE) for image segmentation and unsupervised fire detection. Assessing with a dataset from real-world fires, results showcased our method's superiority. Compared to YOLOv8, our framework boosted precision, recall, and F-score by 19.5%, 1.2%, and 11.7%. Field tests, deploying a robot with the algorithm in an actual fire scenario, highlighted real-world applicability. These experiments emphasized both method performance and practical deployment potential.

Jin et.al [3] The paper addressed the crucial role of flame area extraction in forest fire detection, emphasizing the challenges of accurate early detection due to fire dynamics and background complexity. Existing deep learning approaches had limitations, such as insufficient feature representation. The proposed ADE-Net introduced a dual-encoding segmentation network with attention-based mechanisms, including attention fusion and multi-attention fusion modules, to enhance feature representation and address class imbalance. The attention-guided enhancement module enriched local features, while a global context fusion module ensured effective multi-scale feature extraction. Experimental results demonstrated ADE-Net's competitive advantage in early fire detection from remote sensing images compared to advanced segmentation models.

Kuznetsov et.al [4] The recent surge in numerical fire modeling unveiled insights into building fire safety and code performance. High-fidelity fire simulation, although expensive and complex, prompted the exploration of artificial intelligence (AI) applications for building fire safety design..This facilitated performance-based design and review processes, offering accurate predictions for the response time of ceiling-mounted heat detectors and sprinklers in dynamic fire scenarios. The AI tool also evaluated fire performance in large-open building spaces and rapidly identified design limits. The proposed AI design approach holds the potential for continuous upgrades to address a broader range of building fire scenarios, ultimately achieving intelligent building fire safety design.

Ren et.al [5] The recent focus on utilizing Unmanned Aerial Vehicle (UAV) imagery for forest fire object detection witnessed significant progress. However, existing object detection models often overlooked the exploration of relationships among positive sample features, crucial for robust and representative feature learning. In response, FCLGYOLO was proposed to enhance object information in feature maps. It introduced a Feature Invariance and Covariance Constraint (FICC) structure to maintain feature invariance and eliminate internal correlations among positive samples. Additionally, a Local Guided Global Module (LGGM) enriched object positioning and semantic information in feature maps. Even in challenging scenarios like heavy smoke or tree occlusions, FCLGYOLO outperformed multiple state-of-the-art object detection models on a forest fire dataset, showcasing its superiority.

Schiks et.al [6] Spatial and temporal estimates of burned areas modeled emissions from fire events, considering fire behavior variations over time and space. A method was developed for day-of-burn estimation, using ordinary kriging with satellite-based active fire detection data from MODIS, VIIRS, and their combination. Comparing kriging results, a quasi-validation procedure applied to 37 wildfires in Ontario's boreal forest accurately estimated nearly half of each fire's burned area within one day of occurrence. This approach demonstrated strengths and limitations in mapping individual wildfire progress, emphasizing the need for future validations to address spatial autocorrelation, often overlooked in ecology's day-of-burn analyses.

Liu et.al [7] The paper introduced AEGG-FD, a YOLO fire detection algorithm incorporating an attention-enhanced ghost mode, mixed convolutional pyramids, and flame-centre detection. The enhanced ghost bottleneck stacked to reduce redundant feature mapping, achieving a lightweight backbone with attention for accuracy compensation. A mixed convolution feature pyramid accelerated network inference speed, while the flame-centre detection (FD) module extracted local information for firefighting effectiveness. Experimental results on benchmark fire and video datasets revealed AEGG-FD outperforming classical YOLO-based models (YOLOv5, YOLOv7, YOLOv8), with a 6.5 improvement in mean accuracy (mAP0.5, reaching 84.7%) and 8.4 increase in inferred speed (FPS). Model parameters and size were compressed to 72.4% and 44.6% of YOLOv5, achieving a balanced firefighting model in terms of weight, speed, and accuracy.

Yang et.al [8] This paper explored the application of hyperspectral remote sensing for precise fire monitoring, leveraging its potent capability to capture land surface information. The study introduced a novel fire detection method based on hyperspectral remote sensing, presenting an end-to-end model using a sparse visual transformer. Additionally, a band selection method was proposed within the transformer framework, utilizing sparse attention and top-k selection mechanisms to mitigate the impact of invalid bands in hyperspectral data. A non-maximum attention suppression algorithm and band pruning were integrated for dimension reduction, effectively eliminating invalid and redundant bands. The model employed a band-exclusive-token input mode, aligning pruning operations with band selection. A dedicated hyperspectral fire detection dataset was introduced, validating the proposed model's performance on this dataset.

3.PROPOSED SYSTEM

3.1 Overview

Throughout this research procedure, it's essential to continually evaluate and fine-tune the SVM model's performance on real-world data to ensure its accuracy and reliability in fire detection. This iterative process may involve periodic model retraining to adapt to changing environmental conditions or data distributions. Figure 4.1 shows the proposed system model. The detailed operation illustrated as follows:

Step 1: Image Processing: The research project begins with the acquisition of image data, which can come from various sources such as cameras, drones, or surveillance systems. The image data often needs preprocessing to enhance its quality and prepare it for analysis.

Step 2: SVM Model Building: After preprocessing and feature extraction, the research project involves building a machine learning model, specifically an SVM model. Support Vector Machines are commonly used for binary classification tasks like fire detection. The steps in SVM model building include:

1. **Data Preparation:** Organize the preprocessed image data into a format suitable for machine learning, with labeled samples indicating whether each image contains fire or not.
2. **Feature Vector Creation:** Convert the extracted image features into feature vectors that can be used as input for the SVM.
3. **Training:** Split the dataset into training and validation sets, and use the training data to train the SVM model. The model learns to distinguish between fire and non-fire instances based on the extracted features.
4. **Model Tuning:** Optimize the SVM's hyperparameters (e.g., kernel type, regularization parameters) to achieve the best performance on the validation data.
5. **Model Evaluation:** Assess the SVM model's performance using various metrics like accuracy, precision, recall, and F1-score. Fine-tune the model as needed based on evaluation results.
6. **Step 3: Prediction:** Once the SVM model is trained and fine-tuned, it can be deployed for real-time fire detection. The prediction phase involves:
7. **Real-time Data Acquisition:** Continuously acquire new image data, either through cameras, video streams, or other sources.
8. **Preprocessing for Real-time Data:** Apply the same preprocessing steps to incoming images, ensuring they are in the appropriate format for feature extraction.
9. **Feature Extraction for Real-time Data:** Extract features from the real-time images, just as was done during training.
10. **SVM Classification:** Feed the feature vectors from the real-time data into the trained SVM model for classification. The SVM will determine whether the input image contains fire or not.

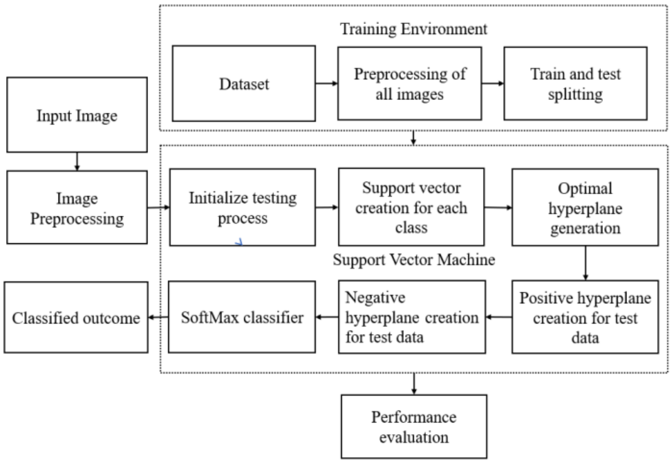

Figure 1 Proposed methodology

3.2 Proposed SVM

SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

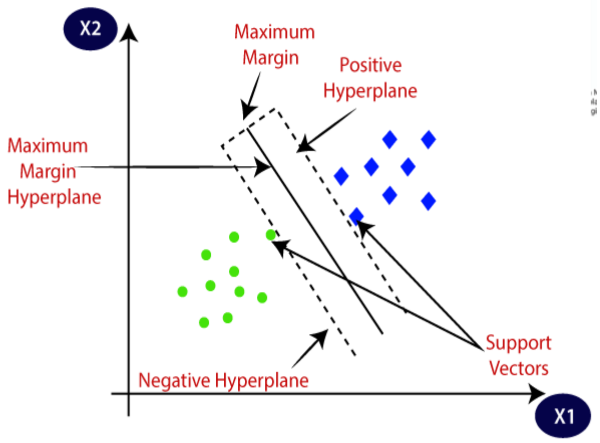

Figure 2 Analysis of SVM

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:

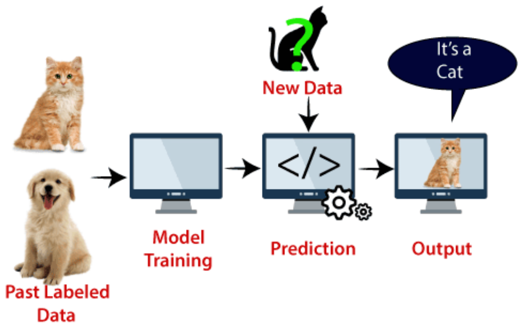

Figure 3. Basic classification using SVM

Types of SVM: SVM can be of two types:

Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier

3.3 SVM working

Linear SVM: The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair (x1, x2) of coordinates in either green or blue. Consider the below image:

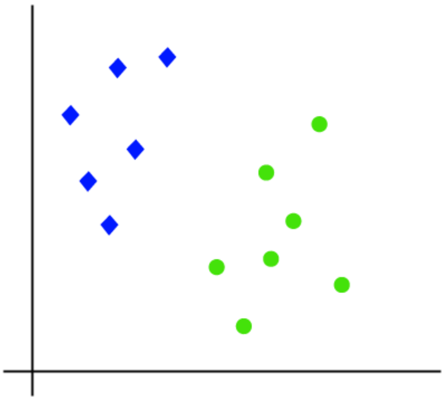

Figure 4. Linear SVM

So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

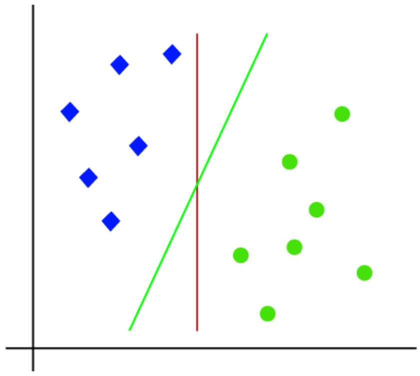

Figure 5. Test-Vector in SVM

Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane.

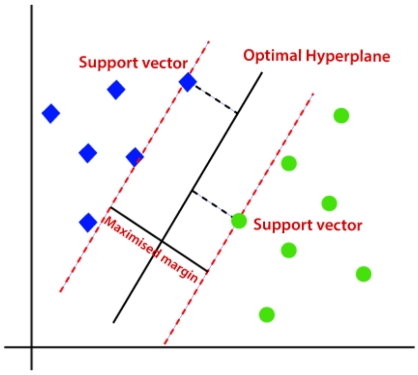

Figure.6. Classification in SVM

Non-Linear SVM: If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:

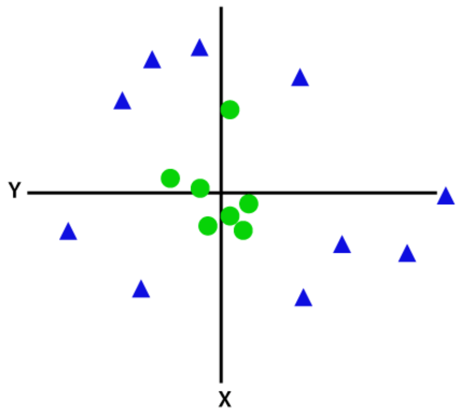

Figure 7. Non-Linear SVM

So, to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third-dimension z. It can be calculated as:

$$z=x^2 + y^2$$

By adding the third dimension, the sample space will become as below image:

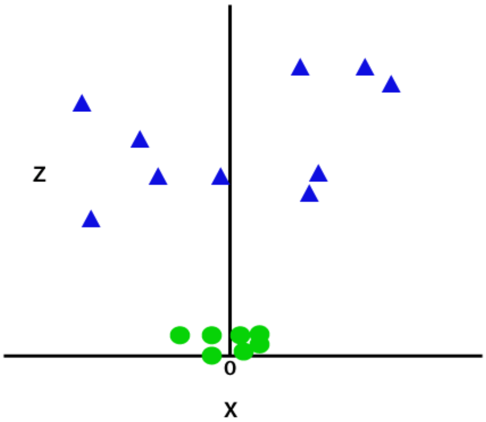

Figure 8. Non-Linear SVM data seperation

So now, SVM will divide the datasets into classes in the following way. Consider the below image:

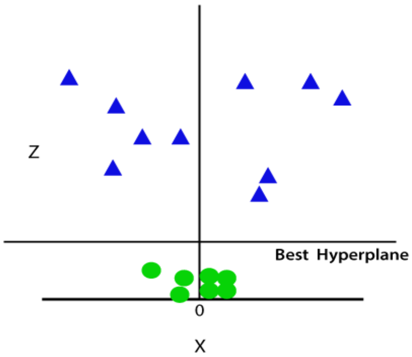

Figure 9. Non-Linear SVM best hyperplane

Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as:

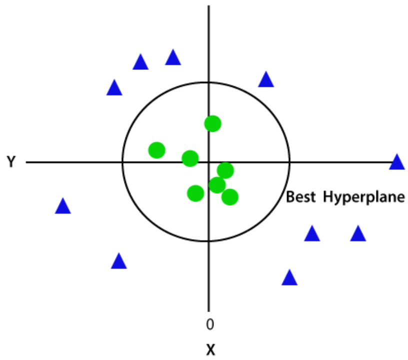

Figure 10. Non-Linear SVM with ROC

4. RESULT

Figure 11 shows a visual representation of a subset of images from the "Normal" class in dataset. These images should serve as examples of what is considered "normal" or non-fire scenarios.

Visualizing samples can help you understand the data distribution and the type of images in this class.

Figure 12 displays a set of images from the "Fire" class in dataset. These images would illustrate instances of fire in various contexts. Visualizing samples from the "Fire" class can provide insight into the variability of fire images in your dataset.

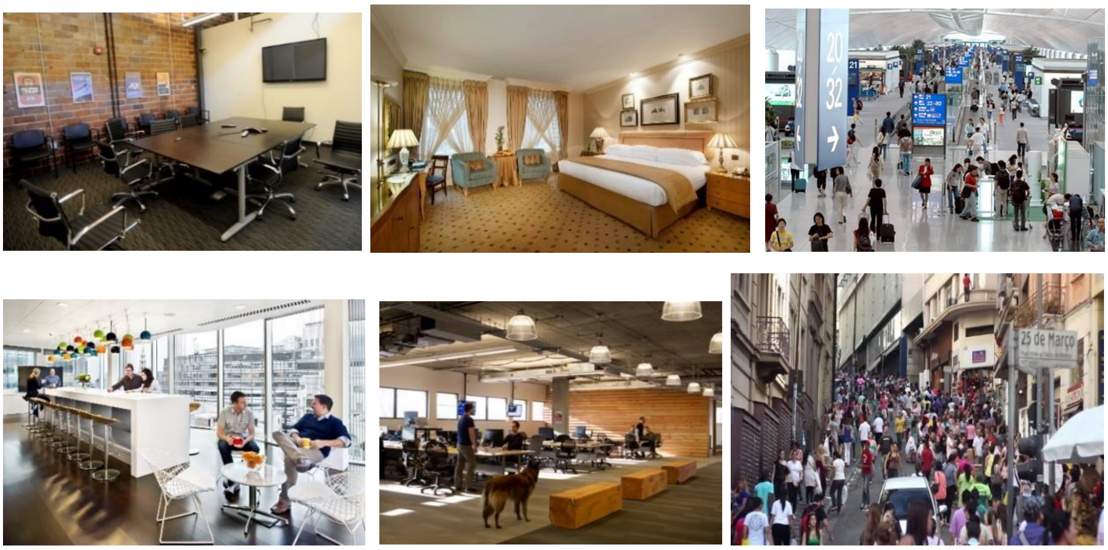

Figure 11: Sample images of dataset with Normal class.

Figure 12: Sample images of dataset with Fire class

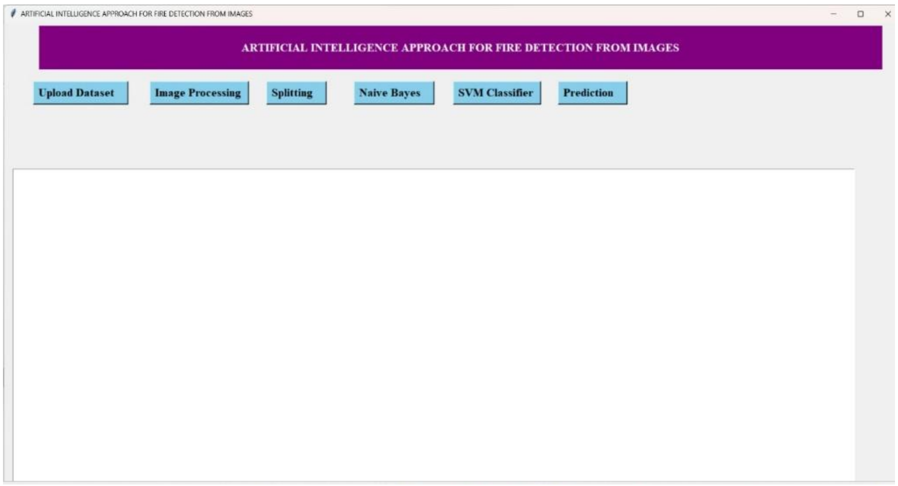

Figure 13 Sample UI used for Fire Detection from images

This figure 13 displays a sample user interface (UI) utilized for fire detection from images. It contains elements such as image upload buttons, processing options, and possibly a display area for the uploaded images.

The figure 15 UI shown here presents categories after uploading a dataset. This involves a list of different classes or labels associated with the images in the dataset.

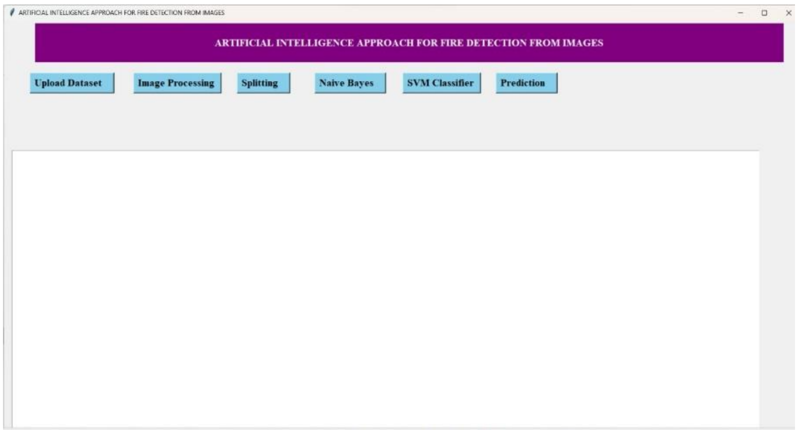

Figure 14 Sample UI used for Fire Detection from images

This Figure 14 UI illustrates the data after image preprocessing. It shows the number of images present in dataset before further analysis or modeling.

The figure 16 displayed here demonstrates the dataset after applying data splitting. This involve dividing the dataset into training, validation, and testing sets for machine learning model training and evaluation.

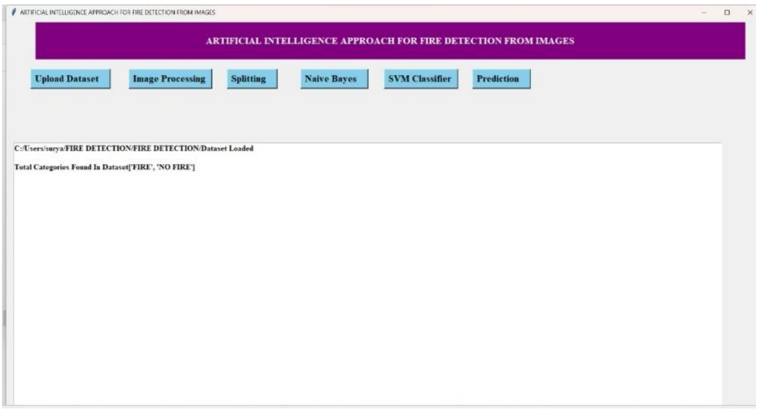

Figure 15: UI shows the Categories after uploading dataset

The figure 18 depicts the performance evaluation metrics of a Naïve Bayes classifier. It includes metrics such as accuracy, precision, recall, and F1-score, among others.

The figure 19 The confusion matrix of the Naive Bayes Classifier is presented in this figure. It shows the actual and predicted classes for the test dataset, facilitating an understanding of the classifier's performance.

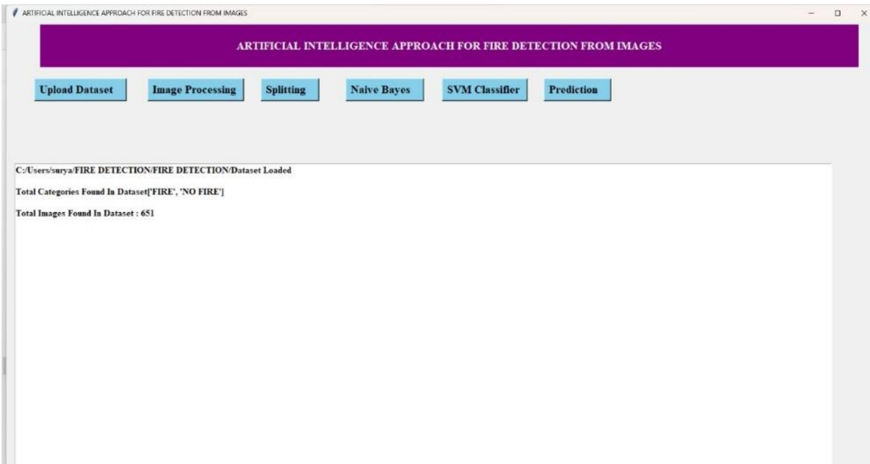

Figure 16 UI shows the Data after image preprocessing

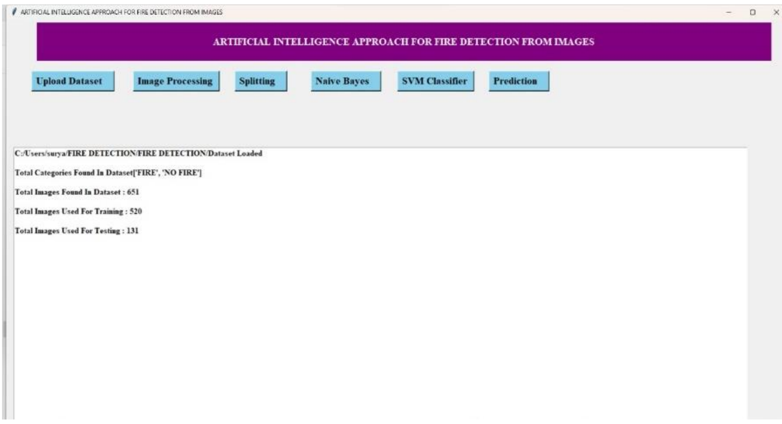

Figure 17: UI shows the dataset after applying data splitting

The Figure 19 shows the performance evaluation metrics, but for a Support Vector Machine (SVM) classifier instead of Naïve Bayes.

The Figure 20 displays the confusion matrix of the SVM Classifier, providing insight into its classification performance across different classes.

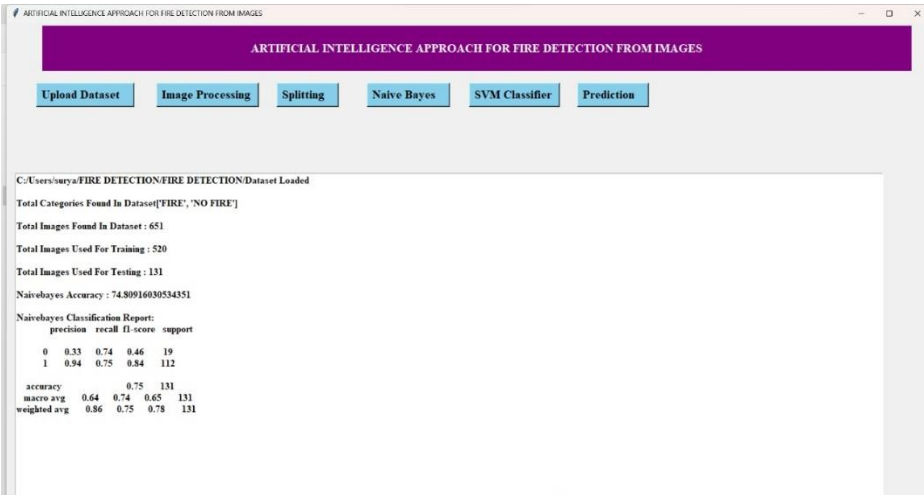

Figure 18**:** Figure shows the performance evaluation of Naïve bayes Classifier

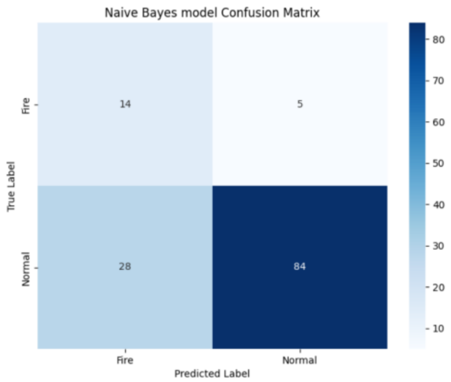

Figure 19: Confusion matrix of Naive Bayes Classifier

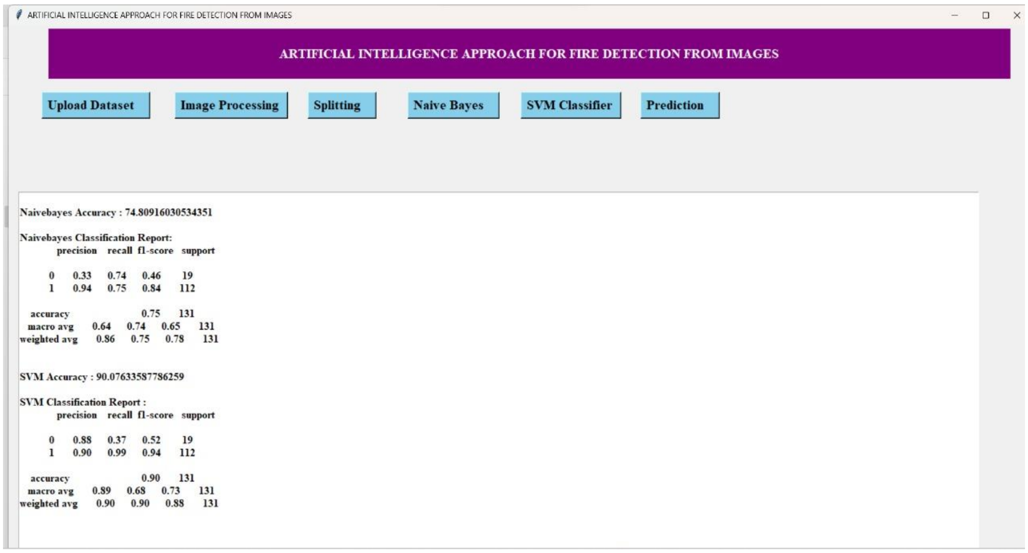

Figure 20: Figure shows the performance evaluation of SVM Classifier

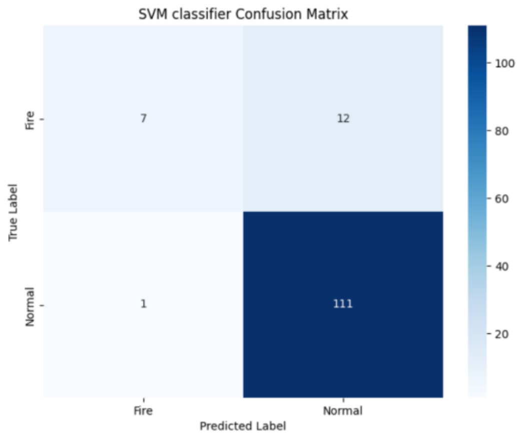

Figure 21: Confusion matrix of SVM Classifier

The Figure 21 Predicted outputs using fire detection from images are shown in this figure. It includes visualizations or textual outputs indicating the presence or absence of fire in the analyzed images based on the employed detection algorithm.

The Figure 22 Predicted outputs using the SVM classifier are depicted in this figure. It showcases the classification results obtained from the SVM model on the test dataset, indicating the predicted classes for each input image.

Figure 22: Predicted output using Fire detection from images

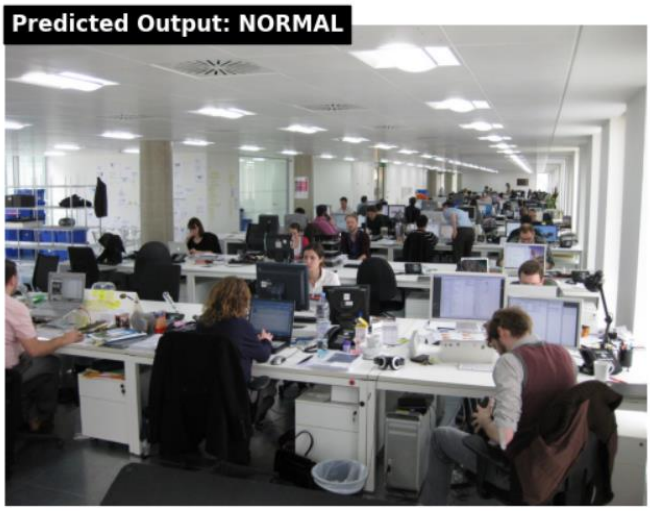

Figure 23: Predicted output using SVM Classifier

- Table 2 provides an overall performance comparison between the Naïve Bayes Classifier and the SVM Classifier. The metrics included are:
- Accuracy (%): This metric measures the overall correctness of the model's predictions. For the Naïve Bayes Classifier, it achieved an accuracy of 75%, while the SVM Classifier achieved a higher accuracy of 90%.
- Precision (%): Precision is a measure of how many of the positive predictions made by the model were correct. In this case, the Naïve Bayes Classifier achieved a precision of 83%, and the SVM Classifier achieved a precision of 85%.
- Recall (%): Recall, also known as sensitivity or true positive rate, measures how many of the actual positive instances were correctly predicted as positive by the model. The Naïve Bayes Classifier had a recall of 72%, while the SVM Classifier had a recall of 87%.
- F1-score: The F1-score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance. The Naïve Bayes Classifier had an F1-score of 85%, and the SVM Classifier also had an F1-score of 85%.
- Based on this table, the SVM Classifier outperforms the Naïve Bayes Classifier in terms of accuracy, while both models have similar precision and F1-score values.
- Table 3 delves into a more detailed comparison of class-wise performance for both the SVM Classifier and the Naïve Bayes Classifier. It evaluates how well each model performs for each class ("Fire" and "Normal") with metrics such as precision, recall, and F1-score.
- For the "Fire" class:
- The SVM Classifier achieves a precision of 73%, indicating that when it predicts "Fire," it is correct about 73% of the time.
- The SVM Classifier has a recall of 36%, meaning that it correctly identifies 36% of the actual "Fire" instances.
- The F1-score for the "Fire" class in the SVM Classifier is 0.45.
- In contrast, the Naïve Bayes Classifier has a lower precision (35%) and a higher recall (73%) for the "Fire" class, resulting in an F1-score of 0.48.

- For the "Normal" class:
- The SVM Classifier has a high precision of 88% and a recall of 97% for the "Normal" class, resulting in an F1-score of 0.48.
- The Naïve Bayes Classifier achieves a precision of 93% and a recall of 72% for the "Normal" class, resulting in an F1-score of 0.81.
- These class-wise performance metrics highlight that the SVM Classifier has a better precision-recall trade-off for the "Normal" class, while the Naïve Bayes Classifier has a better trade-off for the "Fire" class. The choice between the two models would depend on the specific requirements of your application and the relative importance of precision and recall for each class.

Table 2: Overall performance comparison of proposed ML models.

Model name	Accuracy (%)	Precision (%)	Recall (%)	F1-score
Naive bayes Classifier	75	83	72	85
SVM classifier	90	85	87	85

Table 3: Class-wise performance comparison of proposed ML models.

Model name	SVM Classifier		Naive Bayes classifier	
	Fire	Normal	Fire	Normal
Precision	0.88	0.90	0.35	0.93
Recall	0.36	0.97	0.73	0.72
F1-score	0.45	0.48	0.48	0.81

5. CONCLUSION

The integration of AI-enabled cameras for fire detection offers a transformative approach to safety and security, revolutionizing traditional methods. Here is a conclusion based on the search results:In the past, the efficacy of fire and smoke detection was limited by the absence of advanced alert systems, leading to increased fire incidents due to delayed warnings. Traditional detectors, reliant on basic principles like changes in temperature and smoke density, often resulted in false alarms and delayed responses. However, the advent of AI-powered technologies has ushered in a new era of proactive and precise fire detection. By leveraging computer vision and machine learning, AI Video Analytics-Based Smoke and Fire Detection systems can analyze real-time video streams from surveillance cameras with unparalleled accuracy.By harnessing the power of AI Video Analytics, organizations can proactively address potential threats, creating a safer environment. The amalgamation of real-time alerts, advanced analytics, and seamless integration forms a robust safety net, ensuring a rapid response to fire incidents. The future of fire detection lies in the innovative capabilities of AI-enabled cameras, redefining safety standards and safeguarding lives and assets effectivel.

REFERENCES

[1] Zhang, Ziyang, Lingye Tan, and Robert Lee Kong Tiong. "Ship-Fire Net: An Improved YOLOv8 Algorithm for Ship Fire Detection." Sensors 24, no. 3 (22): 727.
[2] Zhao, Hongyang, Jing Jin, Yi Liu, Yanan Guo, and Yi Shen. "FSDF: A high-performance fire detection framework." Expert Systems with Applications 238 (22): 121665.
[3] Jin, Li, Yanqi Yu, Jianing Zhou, Di Bai, Haifeng Lin, and Hongping Zhou. "SWVR: A Lightweight Deep Learning Algorithm for Forest Fire Detection and Recognition." Forests 15, no. 1 (22): 204.
[4] Kuznetsov, G. V., R. S. Volkov, A. S. Sviridenko, and P. A. Strizhak. "Reduction of response time of fire detection and containment systems in compartments." Fire Safety Journal (22): 104089.
[5] Ren, Dong, Yang Zhang, Lu Wang, Hang Sun, Shun Ren, and Jian Gu. "FCLGYOLO: Feature Constraint and Local Guided Global Feature for Fire Detection in Unmanned Aerial Vehicle Imagery." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (22).
[6] Schiks, Tom J., B. Mike Wotton, and David L. Martell. "Remote Sensing Active Fire Detection Tools Support Growth Reconstruction for Large Boreal Wildfires." Fire 7, no. 1 (22): 26.
[7] Liu, Jiansheng, Jiahao Yin, and Zan Yang. "Fire Detection and Flame-Centre Localisation Algorithm Based on Combination of Attention-Enhanced Ghost Mode and Mixed Convolution." Applied Sciences 14, no. 3 (22): 989.
[8] Yang, Lifeng, Feng Yanqing, Yueming Wang, and Jianyu Wang. "Refined fire detection and band selection method in hyperspectral remote sensing imagery based on sparse-VIT." Infrared Physics & Technology (22): 105104.
[9] Anuar, Afiq Ikhwan Mohd, Roslina Mohamad, Arni Munira Markom, and Ronnie Concepcion II. "Real-time forest fire detection, monitoring, and alert system using Arduino." Indonesian Journal of Electrical Engineering and Computer Science 33, no. 2 (22): 942-950.