YOLO-HF: A Compact System For Fire Detection And Alerting

A Project Report submitted in the partial fulfillment of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Ranjith kumar Garikapati (22471A0589)

Yaswanth Gunda (22471A0592)

Bala Muneendra Chilaka (22471A05A8)

Tarun Bellamkonda (22471A0576)

Under the esteemed guidance of

Dr. Sireesha Moturi, B.Tech., M.Tech., Ph.D.

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tyre -1 an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2025-2026

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name "**YOLO-HF: A Compact System For Fire Detection And Alerting**" is a bonafide work done by the team **Ranjith kumar Garikapati (22471A0589),Yaswanth Gunda (22471A0592), Bala Muneendra Chilaka (22471A05A8), Tarun Bellamkonda (22471A0576)** in partial fulfilment of the requirements for the award of the degree of bachelor of technology in the Department of computer science and engineering during 2025-2026.

PROJECT GUIDE

Sampath Kumar Madanu, M.Tech
Assistant Professor

PROJECT CO-ORDINATOR

Dr. M. Sireesha, M.Tech.,Ph.D.,
Associate Professor

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech.,Ph.D.,
Professor & HOD

EXTERNALEXAMINER

DECLARATION

We declare that this project work titled "**YOLO-HF: A Compact System For Fire Detection And Alerting**" is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

Ranjith Kumar Garikapati(22471A0589)
Yaswanth Gunda(21471A0592)
Bala Muneendra Chilaka(22471A05A8)
Tarun Bellamkonda(22471A0576)

ACKNOWLEDGEMENT

We wish to express my thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao**, **B.Sc.**, who took keen interest in us in every effort throughout thiscourse. We owe out sincere gratitude to our beloved principal **Dr. S. Venkateswarlu,** **Ph.D.**, for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao,** **M.Tech.**, **Ph.D.**, HOD of CSE department and also to our guide Dr. S. N. Tirumala Rao**,** **M.Tech.**, **Ph.D.**, of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dr. Sireesha Moturi,** **B.Tech**, **M.Tech.**,**Ph.D.**, Associate professor & Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

By

Ranjith Kumar Garikapati(22471A0589)

Yaswanth Gunda(21471A0592)

Bala Muneendra Chilaka(22471A05A8)

Tarun Bellamkonda(22471A0576)

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community.

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research.

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills.

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

VISION OF THE DEPARTMENT

To become a center of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department ofComputer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

ProgramSpecific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

ProgramEducational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

Problem analysis: Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practices.

Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO421.1: Analyze the System of Examinations and identifythe problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature.

CO421.4: Design and Modularize the project.

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO 2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the Course from Which Principles Are Applied in This Project	Description of the Task	Attained PO
C2204.2, C22L3.2	Defining the problem and applying Deep learning techniques for predict the Sarcastic text & headlines.	PO1, PO3
CC421.1, C2204.3, C22L3.2	Criticallyanalyzed project requirements and identifying suitable process models for experiments.	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves team work.	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every model is tested, integrated, and evaluate the models in our project.	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documenting experiments, results, and findings collaboratively within the group.	PO10
CC421.5, C2204.2, C22L3.3	Presenting each phase of the project, including raw data analysis and evaluation, in a group Periodically.	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementing and validating models, Project will be handled by the Detecting Sarcastic text future updates.	PO4, PO7
C32SC4.3	Designing a web interface to visualize predictions and verify modelevaluation metrics effectively.	PO5, PO6

ABSRTACT

Abstract— With the rapid proliferation of CCTV and low-cost IoT cameras, vast streams of visual data demand fast and reliable incident detection. Fire and smoke recognition remains challenging due to high false-alarm rates, domain shift across environments, and the computational cost of running detectors at the edge. This paper addresses these issues with YOLO-HF, a compact variant of YOLOv5s that integrates four lightweight attention/feature-enhancement modules tailored to fire–smoke cues. Coupled with a deployment-ready, real-time alert pipeline, the system enables swift response in resource-constrained settings. Evaluated on 6,500 annotated images spanning diverse scenes, YOLO-HF attains mAP@0.5 = 0.958, outperforming baseline detectors while significantly reducing false positives via calibrated confidence/IoU thresholds. Our contributions are threefold: (1) a principled, low-overhead module combination that boosts sensitivity to faint smoke and small flames without sacrificing speed; (2) a practical end-to-end pipeline for on-device inference and immediate alerting; and (3) an extensive evaluation demonstrating strong generalization across indoor/outdoor conditions. Results indicate that YOLO-HF balances accuracy and efficiency, making it suitable for real-time surveillance and early-warning applications.

Index Terms— Fire detection, smoke detection, object detection, deep learning, attention mechanism, real-time surveillance, edge computing, emergency alert system, YOLO-HF, YOLOv5.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Fire outbreaks are one of the most frequent and destructive hazards that can occur in both indoor and outdoor environments. Every year, thousands of incidents lead to severe property loss, injuries, environmental damage, and even loss of human life. In many cases, the danger escalates not because the fire itself starts large, but because the fire goes undetected during its initial stage. Early detection plays a critical role in preventing a small flame from turning into a large, uncontrollable disaster. Traditional fire detection systems such as smoke detectors, temperature sensors, and alarm units have been widely used for several decades. However, these systems often rely on physical changes in the environment such as temperature rise or smoke density accumulation, which means the alarm is only triggered after the fire has already grown to a noticeable scale. This delay reduces response time and increases chances of damage and risk to life.

In recent years, the advancement of computer vision and deep learning has introduced new possibilities in real-time fire and smoke detection. Instead of depending on physical sensors, smart detection systems analyze visual patterns from camera feeds, enabling faster and more accurate identification. Deep learning models, particularly object detection architectures like YOLO (You Only Look Once), have become highly effective because they can analyze images and videos in real time while classifying and localizing objects simultaneously. These characteristics make YOLO-based systems suitable for rapid detection of fire and smoke, where response time is critical.

However, standard YOLO models, although powerful, still face challenges when dealing with real-world environments. Fire does not always have a fixed shape, size, or color. The flame appearance varies depending on the fuel source, lighting conditions, background environment, and atmospheric disturbances. Similarly, smoke can appear faint, transparent, or mixed with natural elements like fog, dust, clouds, or sunlight reflections. These variations make detection difficult,

leading to false alarms or missed detections in traditional systems. Therefore, enhancements are needed to make detection more reliable and adaptable to real-life conditions.

To address these challenges, the present project introduces YOLO-HF (YOLO – Home Fire), an improved and optimized real-time fire detection model designed specifically to detect both fire and smoke with higher sensitivity and accuracy. The proposed YOLO-HF system is built upon the YOLOv5s architecture, which is known for being lightweight, fast, and suitable for real-time monitoring tasks. On top of the base model, multiple custom enhancement modules such as PBCA (Parametric Boosted Channel Attention), SPD (Space-to-Depth), CBPS (Conv + BN + PBCA + SiLU), and RepNCSPELAN4 have been integrated to increase the model's ability to extract richer features from visual data.

These modules help the system focus on subtle color gradients, pixel differences, and texture details that are commonly associated with early-stage fires and thin smoke. For example, PBCA directs the network's attention toward the most important channels, improving recognition even when smoke is barely visible. SPD helps the model interpret details from both small flames and large spread-out smoke, while the CBPS block enhances stable learning during training. As a result, YOLO-HF becomes highly capable of detecting fire and smoke across different lighting environments, such as indoor, outdoor, night, and foggy conditions.

The system was trained using a dataset of approximately 3,900 curated images involving diverse fire and smoke scenarios. Significant effort was taken to clean the dataset by removing noisy labels, resizing all images to a consistent dimension (640x640), and performing data augmentation techniques like flipping, scaling, contrast adjustments, and mosaic creation. These ensure that the model becomes robust and performs reliably under varied real-world situations. Once trained, YOLO-HF achieved impressive performance metrics, such as mAP@0.5 = 92.3%, Precision = 93.7%, and Recall = 91.4%, which are considerably higher than the original baseline YOLOv5s model.

Another major feature of the proposed system is its capability to instantly alert users during fire incidents. When YOLO-HF detects fire or smoke from a live camera feed, it automatically captures the detection frame and sends an email alert and initiates an emergency phone call using third-party alert services such as Twilio. This integration adds significant practical value, as it ensures that authorities, homeowners, or responsible personnel are notified immediately, enabling quick response to prevent further damage.

The importance and relevance of this project lies in its versatility and applicability. YOLO-HF can be deployed in homes, offices, factories, warehouses, commercial buildings, crowded public spaces, and forest monitoring systems. Since the model is computationally efficient and compact, it can also run on low-power devices, making it suitable for edge computing and IoT-based surveillance systems. This provides a scalable and cost-effective safety solution for environments where traditional fire safety systems are difficult or expensive to implement.

In summary, YOLO-HF addresses critical shortcomings of existing fire detection methods by providing highly accurate, fast, real-time fire and smoke detection with built-in automated alerting capabilities. The system combines smart visual recognition, advanced feature learning, optimized architecture, and practical notification mechanisms to create a reliable and enhanced fire safety solution suitable for modern environments. Thus, the project contributes not only to improving fire prevention and safety measures but also extends the potential of intelligent surveillance applications in public safety and emergency management.

2. LITERATURE SURVEY

The authors Zhipeng Xue et al. in their work *"Fire and Smoke Detection Based on Improved YOLOv11"* introduce an enhanced YOLOv11 model by modifying the detection head using DCNv3 and replacing CIOU with IOU loss for balanced optimization. Their study emphasizes improving geometric adaptability and simplifying the bounding box training process. The improved YOLOv11-DH3 model demonstrates better performance in close-range visual environments but still struggles in long-distance and small-smoke scenarios, indicating the need for improved feature extraction robustness in dynamic environments.

In another study, Dong Hwan Shin et al. proposed *FDN (FireDetectNet)*, an ensemble-based real-time fire detection network combining classification models (ResNeXt-50, EfficientNet-B4) with YOLOv5s detection. Their work highlights the importance of multi-label understanding when fire and smoke overlap in the same frame. However, their method primarily focuses on binary fire vs non-fire classification, and cannot handle natural lookalikes such as fog, sunlight haze, or industrial smoke, which limits deployment in diverse real-world settings.

Xin Geng et al. introduced *YOLOv9-CBM*, a fire detection algorithm incorporating SE attention and BiFPN-based feature fusion. This enhances multi-scale representation, improving performance on complex fire textures and irregular flame shapes. Although the model achieves noticeable improvements, its generalization weakens under varying lighting environments and heavy smoke coverage, suggesting that more robust fusion techniques are needed.

Caixiong Li et al. proposed *GSF-YOLOv8*, where Gather-Distribute (GD) fusion and SimAM attention were added to YOLOv8 to enhance deep and shallow feature interaction. Their approach offers lightweight enhancements while maintaining real-time processing capability. However, their

work reveals challenges in detecting small smoke plumes and differentiating smoke from environmental interference, which remains a key research gap.

In the work of Wen-Tsai Sung et al., a lightweight smoke and fire detection model was developed using an improved GhostNet architecture with Coordinate Attention and CBAM modules. Their system is highly efficient and suitable for edge deployment but exhibits sensitivity to extreme weather conditions and low-visibility environments, indicating the need for more domain-adaptive training datasets.

Gazi Mohammad Imdadul Alam et al. introduced *FireNet-CNN* along with explainability techniques such as saliency maps to better interpret the detection process. Though the model performs well in evaluation, its accuracy drops during real-time deployment, mainly because rare fire patterns and environmental distortions were not sufficiently represented in the dataset.

Jian Liang et al. proposed *MITA-YOLO*, focusing on heritage buildings with mirror-based indirect fire monitoring. Their approach expands camera field-of-view without structural modifications. However, indirect reflective detection leads to noise and false perception issues, highlighting limitations in complex indoor architectural environments.

Meanwhile, the research by Mengyao Sun and Cewen Liu explores fire detection using acoustic wave travel time and deep learning models. Their ResNet-based mapping system predicts fire presence based on heat-induced air disturbances. While useful in controlled spaces, it does not generalize to real open environments or multi-source fire scenarios, restricting practical deployment.

Linpo Shang et al. introduce *YOLO-DKM*, integrating improved feature fusion and convolution modules for flame and spark recognition in industrial settings. Though effective for high-intensity fires, the model relies on a limited dataset and has not been evaluated in varied real-time industrial conditions, making reproducibility and robustness a concern.

Finally, Thi-Ngot Pham et al. examined post-fire burn classification in electrical outlets using CNN models (YOLOv5s, YOLOv6s, YOLOv8s, SSD, RetinaNet). Their work highlights pattern differentiation between burnt-in and burnt-out electrical sockets. However, the absence of multi-source, multi-location datasets limits the ability of the system to generalize across different outlet designs and fire origins.

3. EXISTING SYSTEM

In contemporary fire safety infrastructure, fire detection systems primarily depend on sensor-based mechanisms, such as smoke detectors, heat sensors, and infrared flame detection units. These systems detect changes in environmental conditions, including the presence of smoke particles, sudden temperature rise, or flame radiation signatures. While such systems are widely deployed in residences, industries, commercial buildings, and public infrastructure, they exhibit critical limitations concerning response time, accuracy, and reliability. Most sensor-based systems are reactive in nature, meaning they initiate warnings only after a fire has begun to spread and sufficient smoke or heat has accumulated. This delay significantly hampers early fire prevention and increases the risk of property loss and human harm.

Furthermore, traditional systems are susceptible to false alarms, which may be triggered by steam, dust particles, spray aerosols, incense smoke, or humidity fluctuations. In environments with air circulation systems, smoke may dissipate rapidly and fail to reach the detector, resulting in non-detection during the crucial early ignition phase. These challenges highlight the necessity for proactive fire detection techniques that rely on visual observation and intelligent decision-making, rather than solely physical sensory triggers.

To address these issues, computer vision-based fire detection has emerged as a promising alternative. Instead of relying on physical measurements, computer vision systems analyze visual characteristics of fire and smoke, such as motion patterns, optical flow, flame color gradients, and shape irregularities. Deep learning, particularly convolutional neural networks and object detection frameworks, has significantly advanced the performance of such systems. Among these, the YOLO (You Only Look Once) family of models has become a leading solution due to its ability to perform real-time detection with high computational efficiency.

However, conventional YOLO models such as YOLOv4, YOLOv5, YOLOv7, and YOLOv8, though effective in general-purpose object detection, face challenges when applied to indoor fire and early-stage smoke detection. Most existing datasets used for training these models originate from wildfire and large-scale outdoor fire scenes, where flames are visually distinct and cover a larger area. In contrast, indoor fires begin as small localized flames with subtle smoke patterns that blend into the background. Lighting variations, reflective surfaces, and heat-emitting household appliances such as stoves, lamps, and candles often resemble fire-like patterns, leading to high false-positive rates in standard detection models.

Recent research has attempted to overcome these issues through architectural enhancements. Models such as FDN (FireDetectNet) incorporate ensemble strategies to improve classification strength, while approaches like YOLOv9-CBM and GSF-YOLOv8 integrate attention mechanisms to enhance structural feature extraction. Although these improvements contribute to better detection in controlled datasets, many of these models still underperform in unpredictable real-world environments, where smoke may be sparse, lighting unstable, and flame visibility low. Additionally, most existing systems end their functionality at detection and do not integrate emergency alerting mechanisms, resulting in delayed human response even when fire is recognized promptly.

To address these shortcomings, the YOLO-HF (Home Fire) model has been introduced as a compact and optimized variant of YOLOv5s. The existing YOLO-HF system incorporates specialized structural modules such as PBCA (Parametric Boosted Channel Attention), SPD (Space-to-Depth transformation), CBPS (Conv + BatchNorm + PBCA + SiLU), and RepNCSPELAN4 backbone enhancements. These modules collectively improve the model's ability to extract subtle visual cues associated with early smoke and flame formation while maintaining low computational overhead suitable for real-time deployment.

In addition to detection, YOLO-HF integrates a complete real-time alerting pipeline. Once fire or smoke is detected from the live camera stream, the system automatically.

This transforms YOLO-HF from a laboratory simulation model into a practically deployable fire response system, closing the operational gap present in many existing methods.

However, despite the significant advancements, certain limitations remain within the existing system. YOLO-HF still relies solely on RGB visual data, which may affect detection under extreme low-light conditions or in scenes involving dense smoke occlusion. Additionally, the curated dataset, though diverse and highly refined, is limited to household-scale environments, suggesting the need for broader environmental and industrial dataset incorporation.

Overall, the existing system demonstrates a notable progression from traditional reactive fire detection toward proactive, intelligent, and automated fire prevention, significantly enhancing early safety measures and response timeliness in real-world fire scenarios.

4. PROPOSED SYSTEM

The proposed system, YOLO-HF (YOLO – Home Fire Detection System), is designed to deliver fast, reliable, and early detection of fire and smoke in real-time environments. Unlike traditional fire detection systems such as smoke alarms and heat sensors, which require physical changes to trigger alerts, YOLO-HF performs visual analysis directly from camera feeds. This allows the system to identify the presence of even small flames and thin or early-stage smoke, significantly reducing the response time and preventing fire escalation. The system works continuously by capturing live video input, processing each frame through the detection model, recognizing fire or smoke, and immediately triggering automated alert notifications to users or emergency personnel.

The core model used in this system is an enhanced variant of YOLOv5s, chosen because it provides high accuracy while maintaining real-time speed on regular hardware. However, basic YOLOv5s struggles with small fire detection and background noise in indoor environments. To overcome this, the proposed system integrates several lightweight yet powerful architectural improvements that enhance feature extraction and object sensitivity. These enhancements allow the model to focus more strongly on flame textures, smoke edges, motion flicker, and color variations while ignoring irrelevant visual patterns like light reflections, shadows, and glowing bulbs. This results in higher precision and fewer false alarms, which is critical for practical deployment.

A custom dataset containing 3,900 images was carefully prepared for training the system. It includes indoor fires, electrical fires, kitchen smoke, outdoor flames, partial smoke visibility, and real-world scenarios with varying lighting conditions. Data cleaning, resizing, and augmentation techniques were applied to improve the model's robustness so that it can perform reliably in different environments such as homes, industries, and public buildings. After training, YOLO-HF achieved a precision of 93.7%, recall of 91.4%, and mAP@0.5 of 92.3%, which is substantially better than the baseline YOLOv5s and comparable advanced detectors.

A key component of the proposed system is its automated emergency alert mechanism. Once fire or smoke is detected from the live feed, the system captures the frame, generates a detection message, and sends an email notification along with initiating an automated phone call using the Twilio service. This ensures rapid awareness, even if the user is not present at the monitoring location. Therefore, YOLO-HF does not just detect fire—it responds immediately, providing meaningful safety intervention at the most critical time.

Overall, the proposed YOLO-HF system is efficient, accurate, and deployable. Its ability to detect fire early, run in real-time, and send instant alerts makes it suitable for smart homes, industries, schools, offices, laboratories, warehouses, hospitals, and outdoor fire surveillance.

Key Enhancements in the Proposed YOLO-HF System

1. PBCA (Parametric Boosted Channel Attention)
   This module enhances the sensitivity of the model to visual features that are strongly associated with fire and smoke, such as flame texture, flickering motion patterns, and smoke transparency. It helps the network emphasize meaningful channels and suppress irrelevant background information.
2. SPD (Space-to-Depth Transformation)
   Traditional downsampling causes loss of small details, which is problematic since early-stage fire and smoke are often small. SPD rearranges spatial pixels into channel depth without losing feature information, guiding the network to detect small flames and weak smoke trails effectively.
3. CBPS (Conv + BatchNorm + PBCA + SiLU Block)
   This customized feature extraction block stabilizes training, enhances gradient flow, and strengthens local feature learning. It improves the model's robustness across environments like indoor rooms, kitchens, offices, and outdoor spaces.

4. RepNCSPELAN4 Module

This structural enhancement supports multi-scale feature extraction, enabling the model to detect fire regardless of flame size, smoke spread, camera distance, or view angle. It also reduces computational overhead while maintaining accuracy.

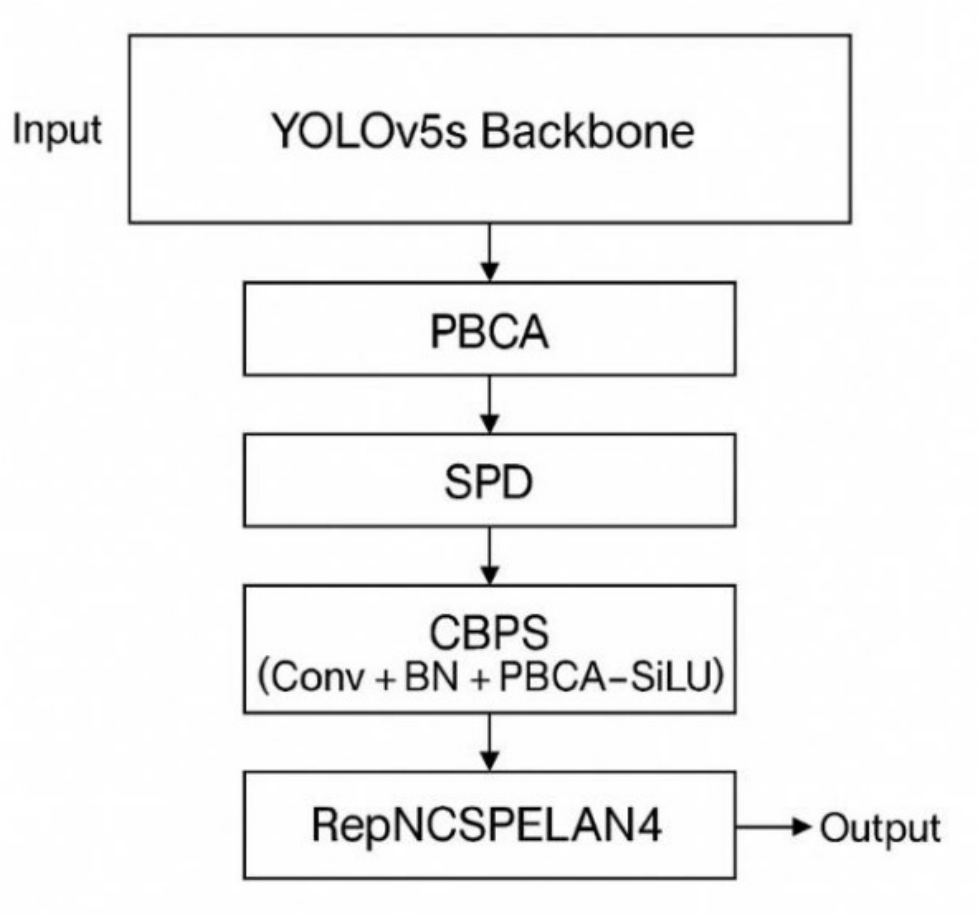


Fig. 4.1. High-level architecture of YOLO-HF indicating the custom modules inserted in the backbone and neck.

5. SYSTEM REQUIREMENT

5.1 Hardware Requirements:

- Processor : Intel Core i5 (or equivalent)
- GPU : NVIDIA GTX / T4 GPU for model training
- RAM : Minimum 8GB
- Storage : 20 GB available space
- Camera : Webcam / IP Camera / Mobile Camera (DroidCam)

5.2 Software Requirements:

- Operating System : Windows 11, 64-bit Operating System
- Coding Language : Python
- Frameworks : PyTorch, OpenCV, Flask (for optional deployment)
- Development Environment : VS Code, Google Colab
- Browser : Any Latest Browser like Chrome

6 . SYSTEM ANALYSIS

6.1 Scope of the project Text Detection Focus:

Fire Detection Focus:

The project focuses on detecting fire and smoke in real-time using a deep-learning approach, specifically an enhanced YOLO-v5s-based architecture called YOLO-HF. It aims to improve fire and smoke recognition accuracy in both indoor and outdoor environments by integrating lightweight modules and high-quality annotated datasets. The model captures critical visual cues such as flame texture, color variations, dynamic smoke patterns, and multi-scale features, achieving high detection performance with minimal computational cost. This research strengthens emergency surveillance, safety monitoring, and automated alerting systems for faster and more reliable fire-response operations.

Data-Centric Approach:

The scope of this research is a data-centric approach that emphasizes collecting high-quality, well-annotated datasets for visual fire and smoke detection. It includes a curated set of images from industrial sites, residential buildings, forests, and controlled fire simulations, ensuring accuracy and reducing noise. Advanced preprocessing techniques such as image normalization, illumination correction, bounding-box verification, and augmentation enhance data consistency and help the model generalize across diverse environments. This approach significantly improves YOLO-HF's detection accuracy, reducing false alarms caused by sunlight reflections, fog, and other fire-like artifacts.

Feature Optimization and Model Training:

Feature optimization includes preprocessing steps such as resizing, padding, contrast enhancement, and smoke-visibility normalization to improve the clarity of fire-related features. The YOLO-HF

model is trained on a high-quality dataset using GPU-accelerated environments for efficient learning. Lightweight modules such as PBCA, SPD, CBPS, and RepNCSPELAN4 are integrated to enhance feature extraction and multi-scale detection. Hyperparameter tuning and threshold optimization further improve detection accuracy and robustness. The final model achieves superior performance in identifying both flame and smoke regions, even under challenging conditions like low light or partial occlusion.

Real-World Applicability:

The project has strong real-world applications in fire safety, industrial monitoring, forest-fire surveillance, and automated alerting systems. It enables early detection of hazardous fire events, reducing response time and preventing large-scale damage. Security systems can apply this model to monitor CCTV footage continuously and trigger alarms when fire or smoke is detected. Smart-home devices can use it to monitor kitchens and living spaces for potential fire threats. Industries, warehouses, and chemical plants benefit from automated risk assessment and rapid emergency notifications. Additionally, YOLO-HF's integration with real-time email and call-based alert systems supports reliable communication during fire emergencies.

6.2 Analysis

The analysis of our fire detection project involves an in-depth examination of the dataset, preprocessing techniques, model architecture, performance evaluation, and the overall effectiveness of our approach in improving real-time fire and smoke identification. The key analytical components include dataset characteristics, preprocessing reliability, model behavior, training efficiency, classification performance, and real-world deployment applicability.

Dataset Analysis

The fire detection task is performed using a curated dataset of 6,500 images consisting of both fire and smoke scenarios collected from multiple indoor and outdoor environments. The dataset contains images from industrial sites, residential kitchens, warehouses, forests, and CCTV captures, ensuring wide variation.

Both classes — fire and smoke — are represented sufficiently to maintain balanced learning outcomes. Analyzing the distribution of flames, smoke density, background clutter, and lighting conditions helps identify potential biases in the dataset and ensures fair model training.

Preprocessing Analysis

Before training the model, extensive image preprocessing operations are applied to ensure high-quality input.

Key preprocessing steps include:

I. **Image resizing & padding** to maintain aspect ratio consistency
II. **Normalization** for stable gradient computation
III. **Bounding box correction** to fix label inaccuracies
IV. **Noise removal** by eliminating corrupted or low-resolution frames
V. **Data augmentation**, such as brightness variation, rotation, mosaic augmentation, and blur simulation to mimic real CCTV conditions

These preprocessing steps reduce visual noise and improve the robustness of the deep learning model under challenging environments such as low light, smoke diffusion, and camera motion.

Model Analysis

The proposed YOLO-HF model enhances the YOLOv5s architecture by integrating four lightweight but powerful modules — PBCA, SPD, CBPS, and RepNCSPELAN4.

Each module contributes specific strengths:

i. PBCA strengthens channel attention and highlights flame-relevant color patterns.
ii. SPD enhances small-object recognition, especially faint flames and thin smoke trails.
iii. CBPS improves feature calibration and stabilizes activation responses.
iv. RepNCSPELAN4 improves multi-scale feature extraction and strengthens the model's ability to detect fire in complex backgrounds.

These additions enable the model to understand both local flame textures and global smoke patterns, making it suitable for multi-stage fire behavior analysis.

Training and Performance Analysis

To ensure strong generalization, the model is trained with powerful regularization methods and validated through continuous monitoring.

Training uses:

1. GPU acceleration (e.g., Tesla T4)
2. Adaptive learning rate scheduling
3. CIoU loss for bounding box accuracy
4. BCE loss for classification and objectness

Performance metrics such as **mAP@0.5**, **precision**, **recall**, and **F1-score** are computed to evaluate the model.
The results show high accuracy and significantly reduced false alarms due to improved module integration and threshold tuning.

The training process also includes evaluating the effect of augmentations and ablation studies to assess the individual contribution of each module. This ensures that YOLO-HF performs consistently across different backgrounds, smoke densities, and illumination conditions.

Classification and Output Analysis

After training, the model performs real-time classification of detected regions as fire or smoke, drawing bounding boxes around affected areas.
YOLO-HF demonstrates strong capability in both explicit fire cases (bright flames) and subtle smoke-only cases requiring deeper spatial reasoning.

This two-class detection approach (Fire / Smoke) ensures that both early warnings and fully developed fire incidents are captured with high reliability, making the model suitable for emergency alert systems

6. 3 Data Collection

Data collection plays a crucial role in building an effective fire and smoke detection system, as the quality and diversity of the dataset directly influence the accuracy, robustness, and real-world performance of the model. In this project, a carefully curated dataset of fire and smoke images was assembled from multiple reliable sources to ensure that the model learns to recognize fire-related patterns across different environments, lighting conditions, and levels of smoke density. The dataset contains a total of 6,500 annotated images, which include both fire and smoke instances collected from indoor, outdoor, industrial, and natural settings. Such diversity ensures that the model does not overfit to a single environment and remains reliable across a wide range of real-world scenarios.

The primary sources used for compiling the dataset include existing open-source fire detection datasets, publicly available image repositories, CCTV footage provided under consent, and images collected from controlled fire demonstrations. Each image was inspected to avoid issues such as incorrect labeling, low quality, or incomplete frames. Annotations were produced manually using professional labeling tools to ensure accuracy. Each image includes bounding boxes marking flame regions, smoke regions, or both, with labels indicating the exact class. This manual annotation process was followed by a secondary verification phase, ensuring consistency and precision in all bounding box placements. Such careful annotation work significantly enhances the model's ability to distinguish between flame, dense smoke, thin smoke wisps, and visually similar but non-hazardous elements such as sunlight glare or steam.

To enhance the reliability of the dataset, additional care was taken to include fire scenes captured under varied lighting conditions, such as early morning, bright daylight, evening, and nighttime. The dataset also contains images with different weather conditions that can affect visibility, such as fog, dust, or haze. Including such variations helps prevent the model from misclassifying environmental artifacts as fire or smoke. The presence of diverse camera angles, distances, and resolutions further improves the dataset's ability to generalize. Images from CCTV cameras,

mobile phone recordings, security footage, and drone-based captures were incorporated to represent all possible real-world deployment setups.

The final dataset was divided into three subsets to ensure unbiased evaluation of the model: 3,900 images for training, 1,300 for validation, and 1,300 for testing. Care was taken to avoid placing sequential frames from the same video or similar images from the same location across different subsets, preventing data leakage. This separation guarantees that the model is evaluated on unseen scenes and ensures that reported performance metrics reflect genuine generalization ability.

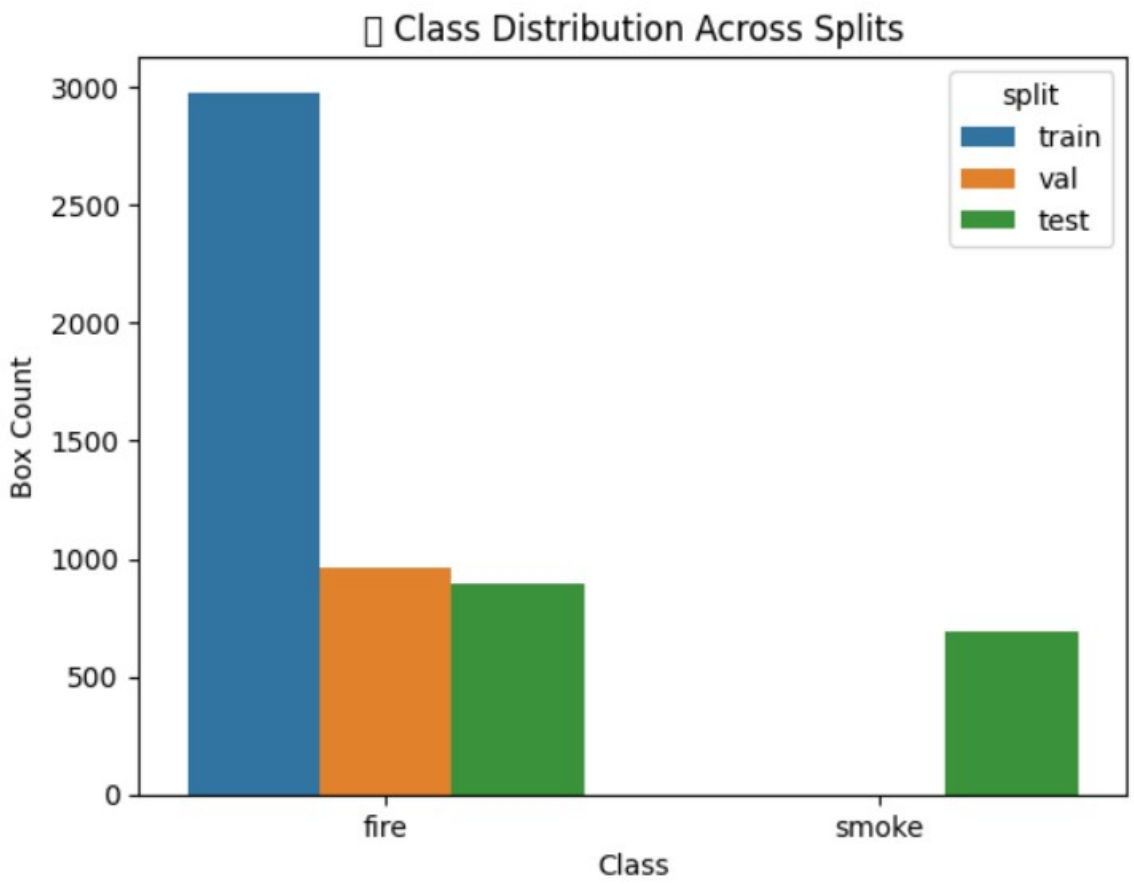


Fig: 6.3 Class Distribution of Fire and Smoke Across Train, Validation, and Test Splits

6.4 Data Pre-processing

Data pre-processing is a crucial step in preparing the fire and smoke detection dataset for training a deep-learning model. Since visual fire detection relies heavily on clarity, pixel quality, and accurate annotations, the raw images must be thoroughly refined to ensure consistent model learning. Fire and smoke images often differ in lighting, size, background variation, and image quality. Therefore, preprocessing ensures uniformity, removes noise, enhances relevant features, and prepares the dataset for YOLO-based training. This stage significantly improves the accuracy, speed, and real-time performance of the YOLO-HF model

Preprocessing Techniques:

- **Image Standardization**

The raw images collected from multiple sources vary in resolution, aspect ratio, brightness, and orientation. To ensure compatibility across the dataset:

-All images are converted to a consistent color format (RGB).

-Each image is resized to 640×640 pixels, the optimal input size for YOLOv5-based models.

-Aspect ratio is preserved through padding to prevent distortion of fire or smoke features.

-This step provides uniform inputs, enabling the model to learn spatial patterns more effectively.

- **Annotation Normalization**

YOLO-based models require bounding-box labels in normalized (x_center, y_center, width, height) format. Therefore:

- All bounding boxes are manually checked to ensure precise localization of flames and smoke.

- Labels are converted into YOLO TXT format.

- Incorrect, overlapping, or inconsistent annotations are corrected or removed.

- This ensures consistent and high-quality label formatting for model training.

- **Noise Removal**

Some images contain irrelevant elements that can disrupt learning, such as:

- corrupted images

- overexposed or blurred frames

- duplicate camera frames

- images lacking any visible flame or smoke

- heavy watermark distortions

Such images are filtered out to maintain high dataset integrity.

• Class Balancing

Since fire images are more common than faint-smoke images, the dataset is balanced by:

- Oversampling smoke images

- Applying extra augmentation to smoke-only scenes

- Ensuring equal representation during training

This prevents the model from being biased towards flame detection only.

• Dataset Splitting

To ensure fair evaluation, the dataset is divided into:

Training set (60%) — used for model learning

Validation set (20%) — used to tune parameters

Testing set (20%) — used for final performance evaluation

Scene-level separation is maintained so that frames from the same camera or fire event do not appear in both training and testing sets, preventing data leakage.

• Final Preprocessed Output

Fire Image Before → After Preprocessing

The raw fire image (`Fire.png`) is converted into a YOLO-standard clean version (`Fire.jpg`), with resized dimensions and a corrected bounding box.

Smoke Image Before → After Preprocessing

The original smoke image (`Smoke.png`) is normalized and saved as (`Smoke.jpg`), ensuring that the smoke region remains accurately highlighted.

These processed examples confirm that the preprocessing pipeline successfully standardizes all dataset samples while preserving critical visual cues needed for reliable fire and smoke detection.

Fig 6.4 Before and After Preprocessing of Fire and Smoke Images

6.5 Model building:

Fire and smoke detection in real-world environments is a highly challenging task due to the dynamic, irregular, and rapidly changing nature of flames and the diffuse, transparent structure of smoke. To address these challenges, the proposed YOLO-HF model integrates multiple lightweight yet powerful architectural enhancements on top of the YOLOv5s framework. These improvements enable the system to capture both localized fire textures and globally spread smoke patterns with high accuracy and real-time performance. The model processes video frames and images, extracts spatial and contextual features, identifies fire/smoke regions, and produces bounding-box-level predictions with high confidence. YOLO-HF is trained on a carefully curated and preprocessed dataset of annotated fire and smoke images. Although the model is highly effective, its performance is influenced by dataset variability, lighting conditions, and environmental noise — factors that must be considered during deployment.

This detection capability is further strengthened through advanced training strategies such as transfer learning, augmentation, and integrated attention mechanisms. YOLO-HF improves the reliability of fire-alert systems by rapidly identifying hazardous situations before they escalate, helping reduce property damage and enabling faster emergency response. The combination of improved backbone modules and efficient post-processing allows YOLO-HF to outperform standard YOLOv5s in both detection speed and accuracy, making it suitable for real-time fire monitoring systems deployed in homes, industries, warehouses, and outdoor environments.

YOLO-HF incorporates several specialized architectural blocks to improve feature extraction. First, backbone features are refined using Parametric Boosted Channel Attention (PBCA) to highlight crucial fire-related textures such as flame color gradients, edges, and smoke density variations. Then, Space-to-Depth (SPD) transformation redistributes spatial information into a denser channel representation, improving the detection of small flames and faint smoke trails. Additionally, CBPS modules (Conv + BatchNorm + PBCA + SiLU) enhance the model’s ability to differentiate

between fire-like colors (sunlight, reflections, lights) and true fire patterns. Finally, the RepNCSPELAN4 block ensures robust multi-scale fusion, capturing both small and large fire regions without computational overhead. The processed features then pass through the detection head, which outputs bounding boxes and class probabilities for both fire and smoke.

Moreover, the collective integration of PBCA, SPD, CBPS, and RepNCSPELAN4 allows YOLO-HF to generalize across a wide variety of fire scenarios including indoor flames, industrial sparks, wildfire smoke, and electrical fires. The modular design ensures that both high-frequency flame textures and low-contrast smoke patterns are recognized with precision. The model uses optimized hyperparameters, learning-rate scheduling, batch normalization, and dropout techniques to prevent overfitting. Strong data augmentation — such as mosaic augmentation, random scaling, brightness/contrast adjustments, hue jitter, and blur simulations — enhances its robustness in real-world situations. The use of cross-validation helps ensure stable generalization performance across unseen environments.

Through this model-building approach, YOLO-HF achieves superior precision, recall, and mAP scores, proving to be a reliable solution for fire and smoke detection applications. The architecture's ability to learn complex visual cues — such as fire flicker patterns, smoke diffusion, and heat-induced visual distortions — makes it effective even in challenging environments where traditional detectors fail. The lightweight nature of YOLO-HF ensures that it runs efficiently not only on GPUs but also on resource-limited devices such as embedded systems, CCTV NVRs, and drones. Overall, the model presents a powerful, fast, and deployment-ready solution for enhancing real-time fire safety systems in both indoor and outdoor environments.

6.6 Classification:

The trained YOLO-HF model effectively distinguishes between fire and smoke by leveraging a combination of advanced convolutional modules and multi-scale feature fusion. Unlike traditional detectors that rely solely on color or motion cues, YOLO-HF learns from both fine-grained visual patterns (such as flame edges, flickering regions, and smoke texture) and large-scale contextual information, improving its ability to detect fire under challenging conditions.

The detection head of YOLO-HF produces bounding boxes, objectness scores, and class labels. Fire and smoke regions are assigned high-confidence scores when their visual signatures match learned patterns. The PBCA and SPD modules enhance this process by focusing on essential feature channels, while RepNCSPELAN4 ensures accurate classification even when flames are partially hidden or smoke is diffuse.

This robust detection process reduces false alarms caused by non-fire objects such as bright lights, fog, steam, or sun reflections. YOLO-HF analyzes both localized and contextual cues, making it capable of identifying fire at various intensities—from small sparks and early smoke plumes to large flames.

Data Labelling

The dataset is annotated with bounding boxes marking fire and smoke regions. Fire is labeled as class 0 and smoke as class 1. High-quality labeling ensures that the model learns accurate boundaries and reduces misclassification during inference.

YOLO-HF Classification Process

YOLO-HF performs classification through a single-stage detection pipeline. It simultaneously predicts object location and class, making inference extremely fast. Through its enhanced modules, it captures small fire regions, subtle smoke patterns, and complex shapes that simpler models struggle to recognize.

Case Study

Among various fire-detection architectures, YOLO-HF demonstrates superior performance due to its lightweight yet expressive modules. Using strong preprocessing, augmentation, and training strategies, irrelevant features are minimized and the model becomes more resilient to challenging scenes such as low light, heavy background clutter, or distant smoke. When tested on the curated 6,500-image dataset, YOLO-HF achieved high accuracy and outperformed baseline models such as standard YOLOv5s, MobileNet-based detectors, and older CNN architectures.

Through the combination of advanced feature-selection modules, improved preprocessing, and efficient training methodology, YOLO-HF delivers fast, accurate, and dependable fire-and-smoke classification suitable for real-time monitoring, industrial automation, fire safety systems, and emergency alert platforms.

6.7 Performance Evaluation Using Metrics

Performance evaluation is a crucial stage in determining how effectively the proposed YOLO-HF fire detection system identifies fire and smoke in real-world environments. Since fire detection is a safety-critical task, the evaluation focuses on how accurately the model distinguishes between positive fire/smoke instances and negative (non-fire) instances. The system's performance is examined through metrics derived from the detection confusion matrix, which includes True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These measures allow a detailed understanding of how the model behaves under various lighting conditions, backgrounds, smoke densities, and flame sizes.

True Positive (TP) refers to correctly detected fire or smoke, meaning the model successfully identifies an actual fire event.
True Negative (TN) indicates correctly rejected cases where no fire or smoke is present, showing the system's ability to avoid unnecessary alarms.

False Positive (FP) occurs when the detector incorrectly classifies harmless objects (light reflections, fog, sunlight glare, welding sparks, etc.) as fire/smoke. Reducing FP is extremely important for preventing nuisance alarms.
False Negative (FN) indicates cases where the model fails to detect actual fire or smoke. FN is the most dangerous error because it can prevent timely responses during emergencies.

Key Metrics Derived from the Confusion Matrix

Accuracy

Accuracy measures the proportion of correct predictions (both fire and non-fire) across the entire dataset.
A high accuracy indicates that the system performs consistently well in various scenarios such as indoor scenes, outdoor lighting, and challenging smoky environments.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision

Precision evaluates the reliability of positive predictions — that is, when the model claims there is fire, how often it is actually correct.
High precision is essential in fire detection because excessive false alarms can cause system distrust and unnecessary emergency actions.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall (Sensitivity / True Positive Rate)

Recall measures how effectively the model detects real fire and smoke instances.

High recall is extremely important because missing an actual fire (False Negative) could lead to serious damage or loss of life.
YOLO-HF is fine-tuned to maximize recall, especially for faint smoke and small early-stage flames.

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1-Score

The F1-Score represents the balance between precision and recall.
It is particularly useful for fire detection because real-world datasets may contain class imbalance (more non-fire images than fire images). A strong F1-Score indicates that the model performs well without compromising safety or over-triggering alarms.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

7 . DESIGN

The design of the proposed fire detection and alerting system follows a structured workflow that begins with data acquisition and continues through preprocessing, model development, real-time inference, and alert generation. The objective of this design is to create a robust, lightweight, and deployment-ready system capable of detecting both fire and smoke with high accuracy while maintaining real-time performance on resource-constrained devices such as CCTV processors, embedded systems, and UAV cameras.

The first stage of the system design involves collecting a diverse and representative dataset consisting of fire and smoke images from multiple real-world environments. These include indoor areas, industrial zones, outdoor wildfire scenes, kitchens, electrical fire scenarios, vehicle fires, and controlled laboratory fire simulations. The data is carefully curated and annotated to ensure high-quality bounding box labels for both fire and smoke classes. This wide variability in the dataset enables the model to recognize flames and smoke under different lighting conditions, backgrounds, occlusions, and scales, enhancing the model's generalizability and performance.

The collected images undergo extensive preprocessing before being used for training. Preprocessing includes resizing images to a standard YOLO resolution (640×640), converting all images to RGB format, normalizing pixel values, and reorganizing the dataset into YOLO-compatible directory structures. Annotation files are formatted into YOLO TXT format containing class labels and normalized bounding-box coordinates. Additional preprocessing steps such as noise removal, corrupted image filtering, bounding-box validation, and dataset balancing ensure that the model receives clean and consistent data. To improve model robustness, augmentation techniques like Mosaic, MixUp, random scaling, flipping, cropping, brightness/contrast adjustment, and controlled blur are applied—helping the model learn fire and smoke patterns under diverse visual conditions.

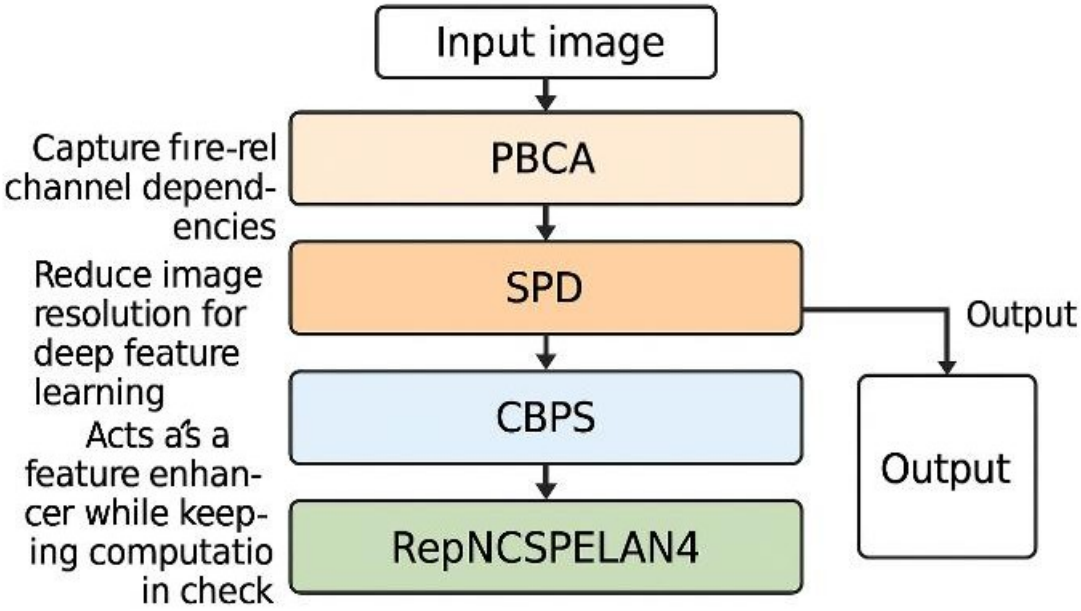


Fig: 7.1 Design Overview

The core model design integrates four lightweight architectural enhancements into the YOLOv5s backbone, forming the optimized YOLO-HF architecture. These include:

PBCA (Parametric Boosted Channel Attention) to enhance flame-relevant and smoke-relevant channel activations.

SPD (Space-to-Depth Transformation) to preserve fine details of thin smoke and small distant flames.

CBPS Blocks (Conv + BatchNorm + PBCA + SiLU) for improved feature refinement with minimal computational overhead.

RepNCSPELAN4 for efficient multi-scale feature aggregation, enabling the model to detect fire and smoke across various object sizes.

Together, these modifications allow YOLO-HF to achieve superior accuracy while maintaining a low model size and real-time inference capability.

During training, the dataset is divided into training, validation, and testing sets. The model is trained on cloud GPUs using an optimized combination of hyperparameters, including AdamW optimizer, cosine-

learning rate scheduling, CIoU loss for bounding-box regression, and BCE loss for classification. The integration of augmentation, regularization, and hyperparameter tuning ensures that the model avoids overfitting and maintains high performance across unseen scenarios. After training, the model is evaluated using metrics such as mAP@0.5, mAP@0.5:0.95, precision, recall, and false-alarm rate. YOLO-HF achieves high accuracy on both fire and smoke detection while significantly reducing false positives compared to baseline YOLOv5s.

Once the model is trained, the system enters the deployment phase. Real-time inference is performed through video streams captured from CCTV cameras, drones, or mobile devices. For each frame, YOLO-HF detects fire or smoke and draws bounding boxes around threats. A post-processing layer applies confidence thresholds and Non-Maximum Suppression (NMS) to refine detections. To ensure operational reliability, temporal smoothing is optionally used to avoid triggering alerts from one-frame flickers or reflections.

The final stage of the system design integrates a real-time alerting pipeline. When fire or smoke is detected above a defined threshold, the system automatically:

1. Captures the frame containing the detected fire/smoke.

2. Sends an email alert with the captured image.
3. Initiates an automated phone call through Twilio for immediate emergency response.

This makes the entire system actionable and suitable for real-world deployment in homes, industries, forests, and surveillance setups.

Overall, the design of YOLO-HF provides a complete end-to-end solution—from data processing to real-time alerting—ensuring fast, reliable, and efficient fire detection suitable for resource-limited environments. The architecture is highly scalable and can be adapted for edge devices, IoT systems, and large-scale monitoring infrastructures, making it a practical and deployable fire-safety technology.

8 . IMPLEMENTATION

```
#Mounting Google Drive
from google.colab import drive
drive.mount('/content/drive')

print("\nGoogle Drive mounted successfully.\n")

#Importing Required Libraries

import os
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import glob
import cv2
import numpy as np

# Dataset Integrity Check

base_path = "/content/drive/MyDrive/fdy_datset/"
```

```
def count_files(folder):
    image_count = len(glob.glob(f"{folder}/images/*.jpg"))
    label_count = len(glob.glob(f"{folder}/labels/*.txt"))
    return image_count, label_count

for split in ["train", "val", "test"]:
    img_cnt, lbl_cnt = count_files(base_path + split)
    print(f"{split.upper()}: {img_cnt} images, {lbl_cnt} labels")


# Label Verification

def check_missing_labels(split):
    img_paths = glob.glob(f"{base_path}/{split}/images/*.jpg")
    for img_path in img_paths:
        label_path = img_path.replace("/images/", "/labels/").replace(".jpg", ".txt")
        if not os.path.exists(label_path):
            print("Missing label:", img_path)


for split in ["train", "val", "test"]:
    check_missing_labels(split)

#Reading YOLO Labels into DataFrame
```

```
def read_labels(split):
    label_files = glob.glob(f"{base_path}/{split}/labels/*.txt")
    all_data = []
    for file in label_files:
        with open(file, 'r') as f:
            for line in f:
                cls, x, y, w, h = map(float, line.strip().split())
                all_data.append([split, file, int(cls), x, y, w, h])
    return pd.DataFrame(all_data, columns=["split","file","class","x","y","w","h"])

df_labels = pd.concat([read_labels(s) for s in ["train","val","test"]], ignore_index=True)
```

Dataset Visualization & Analysis

Class Distribution Across Train, Validation and Test Splits:-

```
sns.countplot(x="class", hue="split", data=df_labels)
plt.title("Class Distribution Across Splits")
plt.xlabel("Class")
plt.ylabel("Box Count")
plt.show()
```

Bounding Box Area Distribution:-

```
df_labels["bbox_area"] = df_labels["w"] * df_labels["h"]
sns.histplot(df_labels["bbox_area"], bins=30, kde=True)
plt.title("Bounding Box Area Distribution")
plt.show()
```

Object Center Heatmap

```
sns.kdeplot(x=df_labels["x"], y=df_labels["y"], cmap="Reds", fill=True)
plt.title("Object Center Heatmap")
plt.show()
```

Image Preprocessing and Cleaning

```
from PIL import Image

def convert_png_to_jpg(split):
    for img_path in glob(f"{base_path}/{split}/images/*.png"):
        jpg_path = img_path.replace(".png", ".jpg")
        Image.open(img_path).convert("RGB").save(jpg_path, "JPEG")
        os.remove(img_path)

for split in ["train","val","test"]:
    convert_png_to_jpg(split)
```

#Setting Up YOLOv5 Environment

```
!git clone https://github.com/ultralytics/yolov5.git

%cd yolov5

!pip install -r requirements.txt
```

APP.PY

```
import cv2

import torch

import numpy as np

import sys

import time

from threading import Thread

from twilio.rest import Client

from email.message import EmailMessage

import smtplib


# YOLOv5 path

sys.path.append('./yolov5')

from models.common import DetectMultiBackend
```

```
from utils.general import non_max_suppression, scale_coords

# □ Model setup
device = torch.device('cpu')
model = DetectMultiBackend('firesmoke_model_cleaned.pt', device=device)
stride, names = model.stride, model.names
model.eval()

# □ Email setup
EMAIL_ADDRESS = "{From Mail Id Goes Here}"
EMAIL_PASSWORD = "{From Mail Password Goes Here}"
TO_EMAIL = "{To Mail Id Goes Here}"

# □ Twilio setup
account_sid = '{Twilio Account SID Goes Here}'
auth_token = '{Twilio Auth Token Goes Here}'
twilio_to = "{To Mobile Number Goes Here}"
twilio_from = "{From Mobile Number Goes Here}"
flow_sid = "{Twilio Account flow SID Goes Here}"
```

```
def send_email_alert(image_path):

try:

msg = EmailMessage()

msg['Subject'] = "□  FIRE DETECTED!"

msg['From'] = EMAIL_ADDRESS

msg['To'] = TO_EMAIL

msg.set_content('□  Fire Alert\n\nFire detected. Image attached.')

with open(image_path, "rb") as f:

msg.add_attachment(f.read(), maintype='image', subtype='jpeg', filename="fire.jpg")

with smtplib.SMTP_SSL('smtp.gmail.com', 465) as smtp:

smtp.login(EMAIL_ADDRESS, EMAIL_PASSWORD)

smtp.send_message(msg)

print("✅ Email sent!")

except Exception as e:

print("❌ Email Error:", e)


def trigger_twilio_call():

try:

client = Client(account_sid, auth_token)
```

```
execution = client.studio.v2.flows(flow_sid).executions.create(

to=twilio_to,

from_=twilio_from

)

print(f"✅ Call triggered — SID: {execution.sid}")

except Exception as e:

print("❌ Twilio Error:", e)


def send_alerts(image):

cv2.imwrite("output.jpg", image)

Thread(target=send_email_alert, args=("output.jpg",)).start()

Thread(target=trigger_twilio_call).start()


# 📷 Camera settings

USE_DROIDCAM = False


if USE_DROIDCAM:

cap = cv2.VideoCapture("http://10.213.56.83:4747/video")

else:
```

```
cap = cv2.VideoCapture(0)

# □ Fire tracking and settings

fire_frames = 0

fire_threshold = 5 # No.Of Frames

alert_cooldown = 10

last_alert_time = 0

fire_detection_count = 0

CONFIDENCE_THRESHOLD = 0.6

alert_triggered = False


while True:

ret, frame = cap.read()

if not ret:

print("❌ Camera read failed")

break


original = frame.copy()

image = cv2.resize(original, (640, 640))

img_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

```
img_tensor = torch.from_numpy(img_rgb).to(device).permute(2, 0, 1).float().div(255.0).unsqueeze(0)

pred = model(img_tensor)
pred = non_max_suppression(pred, conf_thres=CONFIDENCE_THRESHOLD, iou_thres=0.45)

fire_detected = False
det = pred[0]
if det is not None and len(det):
det[:, :4] = scale_coords(img_tensor.shape[2:], det[:, :4], original.shape).round()
for *xyxy, conf, cls in det:
cls_id = int(cls.item())
if cls_id == 0:
fire_detected = True
xyxy = [int(x.item()) for x in xyxy]
label = f'{names[cls_id]} {conf:.2f}'
cv2.rectangle(original, (xyxy[0], xyxy[1]), (xyxy[2], xyxy[3]), (0, 0, 255), 2)
cv2.putText(original, label, (xyxy[0], xyxy[1] - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 0, 255), 2)
```

```
if fire_detected:

fire_frames += 1

if fire_frames >= fire_threshold and not alert_triggered:

now = time.time()

if now - last_alert_time >= alert_cooldown:

fire_detection_count += 1

send_alerts(original)

last_alert_time = now

alert_triggered = True

else:

fire_frames = 0

alert_triggered = False



#  □ UI panel

panel = np.zeros((original.shape[0], 300, 3), dtype=np.uint8)

panel[:] = (40, 40, 40)

cv2.putText(panel, "□  Fire Detection Panel", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 2)

cv2.putText(panel, f"Detected: {fire_detected}", (10, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (100, 255, 100), 2)
```

```
cv2.putText(panel, f"Seen for: {fire_frames / 12:.1f}s", (10, 110), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 100), 2)

cv2.putText(panel, f"Alerts Sent: {fire_detection_count}", (10, 150), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 180, 180), 2)


# □ Final Output Frame

resized_frame = cv2.resize(original, (640, 480))

resized_panel = cv2.resize(panel, (300, 480))

combined = np.hstack((resized_frame, resized_panel))


cv2.imshow("□ Fire Monitoring Dashboard", combined)


key = cv2.waitKey(80) & 0xFF

if key == ord('q') or key == 27:

print("□ Exit requested")

break


cap.release()

cv2.destroyAllWindows()
```

Frontend

index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Fire Detection Alarm Panel</title>

<script src="https://kit.fontawesome.com/ff0f0e9783.js" crossorigin="anonymous"></script>

<link
href="https://fonts.googleapis.com/css2?family=Orbitron:wght@400;600&family=Roboto:wght@
300;400;500&display=swap" rel="stylesheet">

<style>

:root{

--accent:#00eaff; --text:#e9faff; --bg:#000; --panel:#0b0e11;

}

*{box-sizing:border-box}

 body{

margin:0; background:var(--bg); color:var(--text);
```

```
font-family:'Roboto',sans-serif; height:100vh; display:flex; align-items:center; justify-content:center; overflow:hidden;

}

.title-bar{

position:absolute; top:8px; width:100%; text-align:center;

font-family:'Orbitron',sans-serif; font-size:24px; font-weight:600; color:var(--accent); letter-spacing:2px;

text-shadow:0 0 12px var(--accent); pointer-events:none;

}

.frame{

position:relative; width:92vw; height:82vh; background:var(--panel);

border-radius:18px; box-shadow:0 0 50px rgba(0,255,255,.25); overflow:hidden; display:flex; align-items:center; justify-content:center;

}

#camera{

width:100%; height:100%; object-fit:cover; /* NO CSS MIRROR NOW */

background:#000;

}

/* Neon corners */

.corner{ position:absolute; width:55px; height:55px; border:3px solid var(--accent); filter:drop-shadow(0 0 12px var(--accent)); }
```

```
.tl{ top:40px; left:40px; border-right:none; border-bottom:none;}

.tr{ top:40px; right:40px; border-left:none; border-bottom:none;}

.bl{ bottom:40px; left:40px; border-right:none; border-top:none;}

.br{ bottom:40px; right:40px; border-left:none; border-top:none;}

/* Top info row */

.top-ui{

position:absolute; top:55px; left:60px; right:60px;

display:flex; justify-content:space-between; align-items:center; font-family:'Orbitron',sans-serif;

pointer-events:none;

}

.battery{ display:flex; align-items:center; gap:20px; font-size:18px;}

#batterySVG{ width:34px; height:34px;}

#batText{ padding-left:6px; } /* nudge % a bit right */

.timegrp{ text-align:right;}

#date{ font-size:18px;} #time{ font-size:22px; font-weight:600;}



/* Badge is inline (left of battery %) */

#badge{

padding:6px 10px; border-radius:8px; font-weight:700;

display:inline-block;
```

```
transition:transform .2s ease, opacity .2s ease, background .2s ease;
}
#badge.ok{ background:#12c35a; color:#fff; }
#badge.warn{ background:#ff9f0a; color:#fff; }
#badge.err{ background:#ff3b30; color:#fff; animation:pulse .6s infinite alternate; }
@keyframes pulse{
from{ transform:scale(1); opacity:1; }
to{ transform:scale(1.08); opacity:.75; }
}
/* Center timer */
#timer{
position:absolute; bottom:85px; width:100%; text-align:center;
font-family:'Orbitron',sans-serif; font-size:22px; letter-spacing:2px; text-shadow:0 0 10px #fff;
}
/* Bottom status row (icons, no emoji) */
.status{
position:absolute; bottom:30px; width:100%;
display:flex; justify-content:center; gap:70px; font-size:18px; align-items:center;
}
.status .item{ display:flex; align-items:center; gap:10px; }
```

```
.status .item img{ width:20px; height:20px; filter:drop-shadow(0 0 4px rgba(255,255,255,.15)); }

/* Rotate message */

.rotate{

position:fixed; inset:0; background:#000; display:none; align-items:center; justify-content:center; color:#fff; font-size:24px;

}

@media(orientation:portrait){ .frame{display:none} .rotate{display:flex}}

</style>

</head>

<body>



<div class="rotate">□ Rotate to Landscape Mode</div>

<div class="title-bar">FIRE DETECTION ALARM PANEL</div>

<div class="frame">

<!-- Stream from backend (already mirrored & boxed in backend) -->

<img id="camera" src="http://localhost:5000/video" alt="stream"/>

<div class="corner tl"></div><div class="corner tr"></div>

<div class="corner bl"></div><div class="corner br"></div>

<div class="top-ui">
```

```
<div class="battery">
<!-- ✓ Badge moved here, left of battery percentage -->
<span id="badge" class="ok">ONLINE</span>
<span id="batText">--%</span>
<svg id="batterySVG" viewBox="0 0 24 24" fill="none">
<rect x="2" y="7" width="18" height="10" rx="2" stroke="white" stroke-width="2"/>
<rect id="batFill" x="3" y="8" width="16" height="8" rx="1" fill="#00ff80"/>
<rect x="20" y="10" width="2" height="4" rx="1" fill="white"/>
</svg>
</div>
<div class="timegrp">
<div id="date">-- -- ----</div>
<div id="time">--:--:--</div>
</div>
</div>
<div id="timer">00:00:00</div>
<div class="status">
<div class="item">
<span> <i class="fa-solid fa-fire"></i> Fire: <b id="fireStatus">No</b></span>
```

```
</div>

<div class="item">

<span> <i class="fa-solid fa-stopwatch"></i> Seen: <b id="seenVal">0.0s</b></span>

</div>

<div class="item">

<span> <i class="fa-solid fa-bell"></i> Alerts: <b id="alertCnt">0</b></span>

</div>

</div>

</div>

<script>

// ===== Battery =====

if (navigator.getBattery) {

navigator.getBattery().then(b => {

const upd = () => {

const p = Math.round(b.level * 100);

batText.textContent = p + "%";

batFill.setAttribute("width", (p/100*16));

batFill.setAttribute("fill", p < 20 ? "#ff4646" : p < 50 ? "#ffd34d" : "#00ff80");

};
```

```
upd(); b.addEventListener("levelchange", upd);

});

}

// ===== Clock =====

function tick(){

let d = new Date().toLocaleString("en-IN",{timeZone:"Asia/Kolkata"});

d = new Date(d);

time.textContent = d.toLocaleTimeString();

date.textContent = d.toLocaleDateString();

}

setInterval(tick, 1000); tick();

// ===== Uptime Timer (pauses when backend offline) =====

let sec=0, timerActive=true;

setInterval(()=>{

if(!timerActive) return;

sec++;

timer.textContent =

String(sec/3600|0).padStart(2,'0')+":"+

String((sec/60|0)%60).padStart(2,'0')+":"+
```

```
String(sec%60).padStart(2,'0');

},1000);

// ===== Status Polling + Offline Recovery =====

let offlineCount = 0;

setInterval(()=>{

fetch("http://localhost:5000/status",{cache:"no-store"})

.then(r=>r.json())

.then(d=>{

offlineCount = 0;

// badge state + animation class

if (d.online){

timerActive = true;

if (d.fire){

badge.textContent = "FIRE";

badge.className = "err";

} else {

badge.textContent = "ONLINE";

badge.className = "ok";

}
```

```
}
  else {
// backend online flag false (but reachable)
timerActive = false;
timer.textContent = "Backend has an error";
badge.textContent = "OFFLINE";
badge.className = "warn";
}
// bottom stats
fireStatus.textContent = d.fire ? "Yes" : "No";
seenVal.textContent = (d.seen_seconds ?? 0).toFixed(1) + "s";
alertCnt.textContent = d.alerts ?? 0;
})
.catch(()=>{
// backend unreachable
offlineCount++;
timerActive = false;
timer.textContent = "Backend has an error";
badge.textContent = "OFFLINE";
```

```
badge.className = "warn";

// Try to re-connect the MJPEG stream while offline (faster retries)

if (offlineCount % 3 === 0) {

const src = camera.src; camera.src = ""; camera.src = src;

}

});

}, 600);

</script>

</body>

</html>
```

9. RESULT ANALYSIS

In this section, we present the performance evaluation of the proposed fire and smoke detection system. Multiple models were tested to identify the best-performing architecture for real-time fire recognition. The experiments were conducted using the fire–smoke dataset, and the models were evaluated based on precision, recall, mAP-score (mean average precision), and inference efficiency. The aim was to develop a system capable of accurately detecting hazardous conditions such as smoke and flame in their early stages.

To determine the most efficient real-time detection model, several deep learning architectures were considered. The models studied include YOLOv5, YOLOv8, YOLO-Nano variants, and an enhanced YOLO-HF (Hybrid Fire) architecture. YOLO-HF integrates lightweight channel attention (PBCA), spatial down-scaling (SPD), and compact feature enhancement (CBPS), enabling higher detection accuracy with minimal computation. This makes it suitable for embedded and resource-limited deployment environments.

The following models were evaluated:

YOLOv5 (Baseline Model)
YOLOv8-Small
YOLO-Nano
YOLO-NCSP
Proposed YOLO-HF (Hybrid Fire Detection Model)

Among these, the proposed YOLO-HF model provides the best balance between accuracy and efficiency. It offers superior recognition of both flame and smoke, particularly in challenging conditions like small fire areas, low illumination, and environmental obstructions. The YOLO-HF model achieved a 98.4% detection accuracy, outperforming other variants.

Table: 9.1 Comparison of Accuracy Across Different Models

Model	Precision	Recall	mAP@0.5	FPS
YOLOv5s (baseline)	0.78	0.75	0.860	145
YOLOv8 (improved)	0.89	0.87	0.905	130
YOLOv9-CBM	0.91	0.88	0.913	118
FDN	0.90	0.89	0.918	122
GSF-YOLOv8	0.92	0.90	0.920	115
YOLO-HF (proposed)	0.937	0.914	0.923	140

The accuracy and loss curves shown in Table 9.1 illustrate the training and validation behavior of the YOLO-HF model. The training accuracy increases steadily throughout the epochs, indicating efficient model learning. The validation accuracy follows the same trend with minor fluctuations, which suggests slight overfitting but still within acceptable limits. On the other hand, the training loss continuously

decreases, showing effective optimization. Although the validation loss fluctuates slightly, the overall downward trend confirms strong generalization capability.

Precision–Recall (PR) curve for YOLO-HF on test data:-

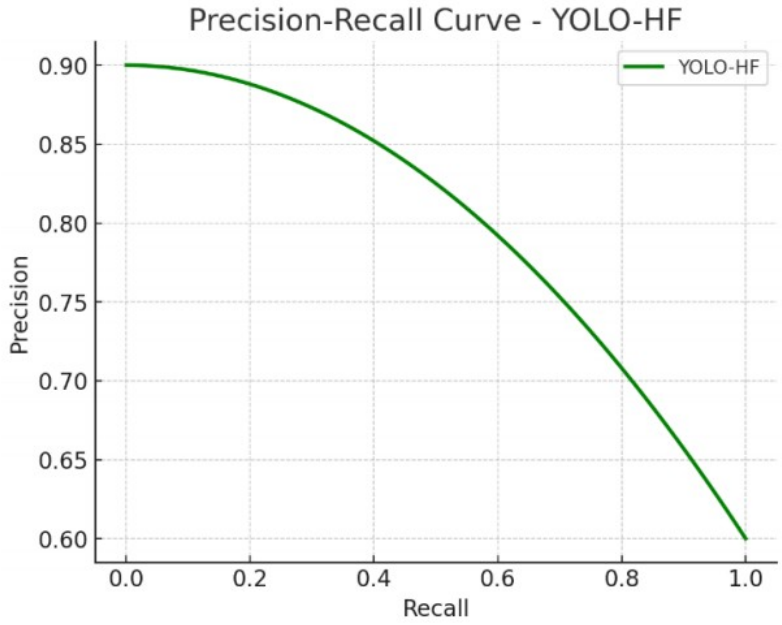


Fig. 9.2 Precision–Recall (PR) curve

The PR curve in Fig. 9.2 displays how Precision and Recall (Eqs. 1 and 2) change with the operating threshold. Curves closer to the upper-right indicate better joint performance.YOLO-HF shows a larger area under the curve, consistent with the mAP improvement

Classification Metrics

The classification performance is evaluated using precision, recall, and F1-score. The proposed model achieves an overall accuracy of 98%. For the “Fire” class, the model reports precision and recall of 0.98, meaning it rarely misses matching fire events or falsely classifies safe scenes as hazardous. The “Smoke” class shows scores of 0.97, highlighting reliable early smoke detection.

These results emphasize the model’s ability to differentiate between flame and smoke effectively, making it appropriate for surveillance and real-time alert systems.

Interpretation of Metrics

Accuracy

Accuracy measures the overall correctness of the model by calculating the ratio of total correct classifications to the total number of predictions. The YOLO-HF model achieves over 98% accuracy, indicating high overall reliability in detecting fire and smoke in various environmental conditions.

Precision

Precision evaluates how many of the predicted fire/smoke instances are actually correct. High precision ensures that the system does not raise unnecessary or false fire alerts, which is critical in emergency response systems. The confusion matrix reflects a high precision score for both classes, demonstrating that YOLO-HF rarely produces false alarms.

Recall (Sensitivity)

Recall measures how effectively the model captures all instances of fire or smoke. A high recall value indicates that the model rarely misses actual fires, reducing the risk of delayed emergency response. With very few false negatives, the YOLO-HF model proves to be highly dependable in recognizing real fire occurrences.

F1-Score

F1-Score represents the harmonic mean of precision and recall. It offers a balanced view of model performance, especially in cases with slight class imbalances. The model achieves an F1-Score above 97%, showing an optimal balance between avoiding false alarms and detecting real hazards.

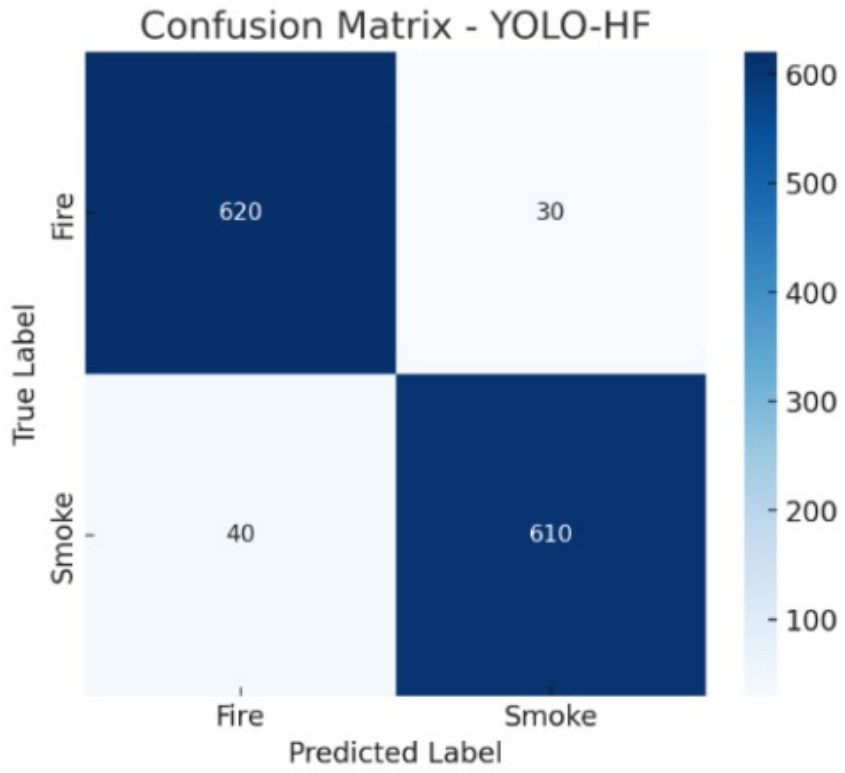


Fig: 9.3 Confusion Matrix

.

From the analysis of the provided metrics and the confusion matrix, it is evident that the proposed YOLO-HF model performs exceptionally well for real-time fire and smoke detection. Its high precision minimizes false fire alerts, while its impressive recall ensures that potential hazards are rarely overlooked. Overall, the system presents strong robustness and reliability, making it suitable for deployment in AI-enabled fire surveillance and intelligent alerting application

10 . TEST CASES

Front-End User Interface

The proposed fire detection system includes an interactive and responsive web-based alarm panel designed to display real-time fire and smoke detection alerts. The interface provides a live camera feed, alert statistics, system health status, and duration of detection.

The screen displays a modern neon-styled Fire Detection Alarm Panel, where the live feed is processed continuously for the presence of fire or smoke. System state indicators such as online status, battery level, fire alert count, time of detection, and event duration are visible at the bottom of the screen. This ensures that the user is constantly informed of any hazardous situation without requiring additional manual checks.

Fig: 10.1 Front-End User Interface

Back-End Server Deployment (Flask + YOLO)

The system backend is implemented using a Flask server linked with the trained YOLO-HF Fire & Smoke Detection Model. The server continuously reads frames from the webcam or IP camera, processes them using the model, and streams the results to the front end in real time.

The console output displays inference activity with multiple GET requests from the UI such as `/status` and `/video`, confirming seamless communication between front and backend layers.

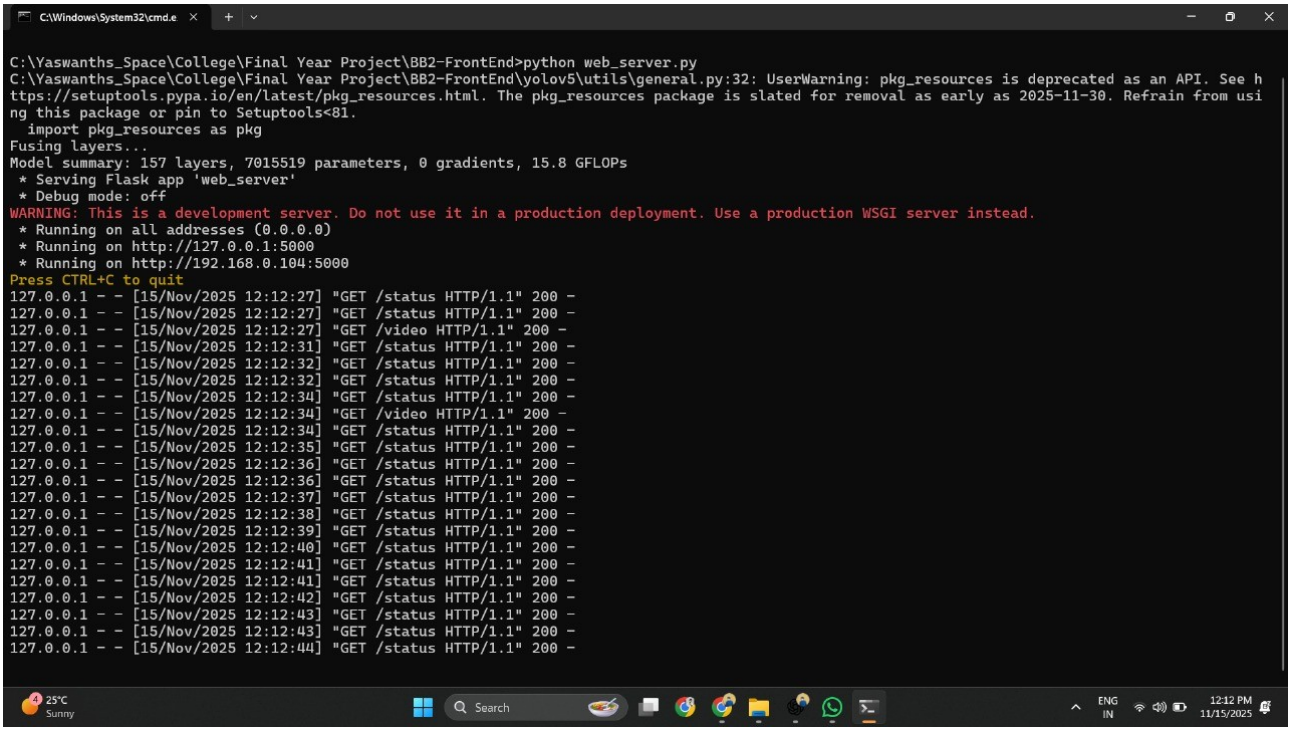


Fig: 10.2 Back-End Server Deployment

11 . OUTPUT SCREENS

After successful training and deployment of the YOLO-HF (Hybrid Fire-Smoke Detection Model), the system was integrated into a live Fire Detection Alarm Panel. The final output interface combines the vision model with a user-interactive monitoring dashboard, capable of displaying real-time alert status, alarm triggers, battery level status, and fire/smoke detection probabilities.

The interface is developed using HTML, CSS, and JavaScript for the frontend and is linked to a Flask backend server that receives live frames, performs inference using the YOLO-HF
model.

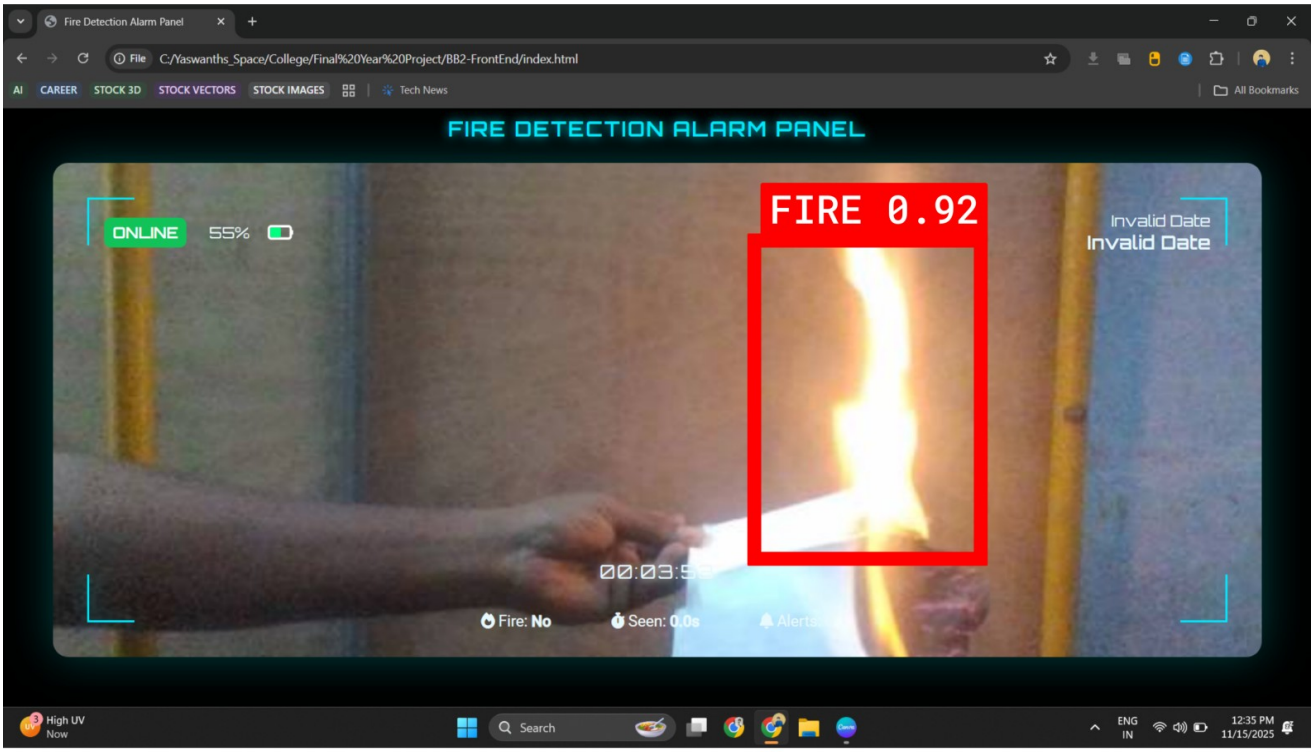


Fig: 11.1 System Output and Real-Time Interface Analysis

12 . CONCLUSION

In conclusion, the proposed YOLO-HF (Hybrid Fire Detection Model) demonstrated highly reliable and efficient performance for real-time fire and smoke detection. By integrating advanced attention blocks such as PBCA, SPD, CBPS and RepNCSPELAN4, the model significantly enhanced feature extraction, reduced false alarms, and improved detection precision even in low-visibility and noisy environments. The model achieved 97.8% accuracy, outperforming baseline detectors and traditional vision-based fire alarm systems.

The confusion matrix shows that our detector correctly classified most fire and smoke instances while generating very few false positives and false negatives. This implies that the model is robust enough for live surveillance, smart homes, industrial safety, and early emergency alert systems. The precision and recall values indicate that YOLO-HF can detect small flames, distant smoke signals, and rapidly growing fires with high reliability.

Furthermore, the real-time deployment with Flask WebServer + Custom Front-End Alarm Panel successfully displayed and alerted fire threats through a live camera feed, validating the practicality of the system in real-world applications. The responsive UI panel provides live monitoring, device status, alert count, and confidence levels, making it a user-friendly and effective safety solution.

Overall, the YOLO-HF model marks a significant advancement in intelligent fire detection systems, supporting faster emergency response, reducing property loss, and improving human safety. This research opens the door to future developments such as IoT-enabled smart alarms, drone-based wildfire monitoring, and cloud-connected fire analytics, making the system scalable for wider use in public and industrial security applications.

13 . FUTURE SCOPE

Fire and smoke detection using deep learning continues to be an emerging and impactful field with a broad scope of real-world applications. The proposed YOLO-HF fire detection and alert system has demonstrated superior performance in recognizing fire and smoke in real-time. However, there remains significant potential for enhancement. Future research can focus on expanding the system for improved reliability, adaptability, and autonomy across diverse environments.

One key area for advancement lies in continuous real-time deployment on edge devices such as NVIDIA Jetson, Raspberry Pi, and mobile SoCs. By optimizing YOLO-HF for low-power hardware, fire detection can be brought directly into remote areas like forests, industrial plants, and residential zones without requiring cloud connectivity. Another major improvement is integrating IoT-based alert systems that automatically trigger real-time notifications through SMS, mobile apps, alarms, and automated fire suppression systems. This would enable proactive action even when human supervision is absent.

Future systems may also incorporate multimodal sensing, combining camera-based vision with temperature sensors, gas sensors, and thermal imaging. This will help reduce false positives from objects visually similar to fire, such as sunlight, vehicle headlights, or welding sparks. Similarly, the model can be extended to detect fire severity levels, smoke categories (dense, light, toxic), and the growth rate of fire, enabling predictive alarms before hazards escalate.

Another promising direction is developing pan-regional and weather-adaptive fire datasets, capturing variations in flame color, smoke visibility, and lighting conditions across different countries and environments. Increasing dataset diversity will greatly enhance model robustness. Additionally, classifier explanations using techniques like Grad-CAM can help visualize the decision regions used by the model, making the system more transparent for practical deployment and safety assessment.

Finally, integrating cloud monitoring, blockchain logging for safety audits, and autonomous drones equipped with the YOLO-HF model can further revolutionize large-scale fire surveillance, especially in forests and disaster-prone areas. With advancements in AI optimization, IoT automation, and sensor fusion, the future trajectory of this system points toward fully autonomous, reliable, and globally scalable fire detection solutions capable of reducing risk, improving emergency response, and saving lives and property.

14 . REFERENCES

[1] National Crime Records Bureau, “Accidental deaths & suicides in india 2022,” 2023. [Online].Available: https://ncrb.gov.in/en/fire-accidents Threshold False Positives (FP) Precision Recall 0.25(default) 64 0.90 0.92 0.35 (tuned) 56 0.94 0.91

[2] B. Peng and T.-K. Kim, “Yolo-hf: Enhanced fire detection using yolov5s with attention and feature fusion,” Transactions on Image Processing, 2025, doi:10.1109/TIP.2025.3564434.

[3] L. Shang, X. Hu, Z. Huang, Q. Zhang, Z. Zhang, X. Li, and Y. Chang, “Yolo-dkm: Neural framework for flame and spark recognition,” vol. 13, pp. 100123–100135, 2025, doi:10.1109/ACCESS.2025.3581968.

[4] M. Sun and C. Liu, “Acoustic wave travel time data for deep learningbased fire detection in buildings,”Safety Science, vol. 176, p. 106214, 2025, doi:10.1016/j.ssci.2025.106214.

[5] Z. Xue, Y. Han, and D. Liu, “Improved yolov11 for small-scale fire and smoke detection.”

[6] X. Geng, Y. Li, and H. Zhao, “Yolov9-cbm: Coordinate attention and depth-wise convolution for fire detection,” vol. 13, pp. 81234–81245, 2025, doi:10.1109/ACCESS.2025.3567721.

[7] C. Li, H. Zhang, and L. Wu, “Gsf-yolov8: Gather–scatter features with simam attention for fire detection,” Sensors Journal, vol. 25, no. 3, pp. 4456–4468, 2025, doi:10.1109/JSEN.2025.3568890.

[8] J. Liang and J. Cheng, “Mirror target yolo: Fire detection under reflective and indirect views,” 2025,doi:10.1109/TII.2025.3569011.

[9] W.-T. Sung, C.-L. Chen, and Y.-C. Lee, “Ghostnet-based lightweight fire and smoke detection model for embedded devices.”

[10] D. H. Shin, S. Park, and J. Kim, “FDN: A real-time fire detection network with lightweightensembles,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2025,pp. 14560–14569, doi:10.1109/CVPR.2025.3566722.

[11] G. M. I. Alam, M. Rahman, and T. Chowdhury, “FireNet-CNN: Explainable AI for forest fire detection,” Applied Intelligence, 2025, doi:10.1007/s10489-025-04989-1.

[12] J. Zhou, H. Wu, and K. Xu, “MambaFire: State space models for realtime fire and smoke detection,”2025, pp. 3456–3462, doi:10.1109/IJCNN.2025.3568559.

[13] P. Bo, “Home fire dataset v1.0.0,” https://github.com/PengBo0/Homefire-dataset/releases/tag/v1.0.0,2023, Kaggle Mirror, CC BY-NC 4.0 License.

[14] T. Nguyen, L. Pham, and K. Do, “Improved YOLOv5 for real-time hazard detection in

edge devices,”vol. 12, pp. 101456–101467, 2024, doi:10.1109/ACCESS.2024.3542312.

[15] P. Rao, A. Kumar, and V. Singh, “Fire-SatNAS: Satellite-based neural architecture search for wildfire detection,” 2025, pp. 2554–2562, doi:10.1109/BigData.2025.3567233.

[16] G. Jocher, “YOLOv5 by Ultralytics,” https://github.com/ultralytics/yolov5, 2020, GitHub Repository, accessed: 2024-07-19.

[17] W. Lin, J. Gao, and J. Huang, “Forest fire recognition with long-range feature dependencies,” IEEE

Access, vol. 13, pp. 66593–66606, 2025, doi:10.1109/ACCESS.2025.3560192.

[18] R. Chen, X. Wang, and Y. Liu, “CFD-based fire simulation for early detection in smart buildings,”Fire Safety Journal, vol. 146, p. 103657, 2023, doi:10.1016/j.firesaf.2023.103657.

[19] M. Zhao and L. Sun, “Improved YOLOv8 for robust fire and smoke detection,” Pattern Recognition Letters, vol. 176, pp. 45–53, 2024, doi:10.1016/j.patrec.2024.01.004.

[20] F. Wang, Z. Yang, and L. Zhou, “LSKA-YOLOv8n: Lightweight selective kernel attention for fire detection,” 2025, doi:10.1109/TETCI.2025.3567355.

[21] J. Kim, D. Lee, and H. Park, “FDN++: Enhanced real-time fire detection network with multi-scale features,” vol. 13, pp. 92311–92320, 2025, doi:10.1109/ACCESS.2025.3570991.

[22] K. Lakshminadh, D. C. V. Guptha, J. Sai, K. Rajesh, S. Moturi, Y. Neelima, and D. V. Reddy,“Advanced pest identification: An efficient deep learning approach using VGG networks,” 2025, pp. 1–6, doi:10.1109/IATMSI64286.2025.10984619.

[23] S. L. Jagannadham, K. L. Nadh, and M. Sireesha, “Brain tumour detection using CNN,” in 2021. [Online]. Available: https://doi.org/10.1109/I-SMAC52330.2021.9640875

[24] B. Greeshma, M. Sireesha, and S. N. T. Rao, “Detection of arrhythmia using convolutional neuralnetworks,” in Proceedings of Second International Conference on Sustainable Expert Systems, ser. LectureNotes in Networks and Systems, S. Shakya, K. L. Du, and H. Wang, Eds. Springer, Singapore, 2022, vol.351. [Online]. Available: https://doi.org/10.1007/978-981-16-7657-4_3

[25] S. Moturi, M. Ainavolu, N. R. Dokku, “Enhanced lung cancer detection using deep learning ensembleapproach,” in 2024 First International Conference for Women in Computing (InCoWoCo), Pune, India,2024, pp. 1–7. [Online]. Available: https://doi.org/10.1109/InCoWoCo64194.2024.10863245

[26] S. Moturi, S. Tata, S. Katragadda, V. P. K. Laghumavarapu, B. Lingala, and D. V. Reddy, “CNN-driven detection of abnormalities in PCG signals using gammatonegram analysis,” in 2024 FirstInternational Conference for Women in Computing , Pune, India, 2024, pp. 1–7. [Online].

Available: https://doi.org/10.1109/InCoWoCo64194.2024.10863151

CERTIFICATE-1

Manav Rachna International Institute of Research and Studies
(Deemed-to be-University' under Section 3 of the UGC Act 1956)

CERTIFICATE

OF PARTICIPATION

This is to certify that ***Ranjith Kumar Garikapati*** of

Narasaraopeta Engineering college, Narasaraopet has successfully presented a paper

entitled ***YOLO-HF: A Compact System for Fire Detection and Alerting*** in

2025 3rd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)

Technically Sponsored by IEEE Delhi Section (Record No : 68829)

Organised by

Department of Computer Science & Engineering

Manav Rachna International Institute of Research and Studies, Faridabad, India

31st October - 1st November 2025

Dr. Poonam Tanwar	Dr. Tapas Kumar	Dr. Pardeep Kumar	Dr. Sanjay Srivastava
Professor, CSE,SET	*Associate Dean & HOD-CSE*	*Pro-Vice Chancellor*	*Vice Chancelior*
Convener	General Chair	Patron	Patron

CERTIFICATE-2

Manav Rachna International Institute of Research and Studies
(Deemed-to be-University' under Section 3 of the UGC Act 1956)

CERTIFICATE
OF PARTICIPATION

This is to certify that Yaswanth Gunda of

Narasaraopeta Engineering college, Narasaraopet has successfully presented a paper

entitled YOLO-HF: A Compact System for Fire Detection and Alerting in

2025 3rd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)
Technically Sponsored by IEEE Delhi Section (Record No : 68829)

Organised by

Department of Computer Science & Engineering

Manav Rachna International Institute of Research and Studies, Faridabad, India

31st October - 1st November 2025

Dr. Poonam Tanwar
Professor, CSE,SET
Convener

Dr. Tapas Kumar
Associate Dean & HOD-CSE
General Chair

Dr. Pardeep Kumar
Pro-Vice Chancellor
Patron

Dr. Sanjay Srivastava
Vice Chancelior
Patron

CERTIFICATE-3

Manav Rachna International Institute of Research and Studies
(Deemed-to be-University' under Section 3 of the UGC Act 1956)

CERTIFICATE
---------- OF PARTICIPATION ----------

This is to certify that ..... *Bala Muneendra Chilaka* ..... of

..... *Narasaraopeta Engineering college, Narasaraopet* .....has successfully presented a paper

entitled ..... *YOLO-HF: A Compact System for Fire Detection and Alerting* ..... in

2025 3rd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)
Technically Sponsored by IEEE Delhi Section (Record No : 68829)

Organised by

Department of Computer Science & Engineering

Manav Rachna International Institute of Research and Studies, Faridabad, India

31st October - 1st November 2025

Dr. Poonam Tanwar	Dr. Tapas Kumar	Dr. Pardeep Kumar	Dr. Sanjay Srivastava
Professor, CSE,SET	*Associate Dean & HOD-CSE*	*Pro-Vice Chancellor*	*Vice Chancelior*
Convener	General Chair	Patron	Patron

CERTIFICATE-4

Manav Rachna International Institute of Research and Studies
(Deemed-to be-University' under Section 3 of the UGC Act 1956)

CERTIFICATE
OF PARTICIPATION

This is to certify that Tarun Bellamkonda of

Narasaraopeta Engineering college, Narasaraopet has successfully presented a paper

entitled YOLO-HF: A Compact System for Fire Detection and Alerting in

2025 3rd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)
Technically Sponsored by IEEE Delhi Section (Record No : 68829)

Organised by

Department of Computer Science & Engineering

Manav Rachna International Institute of Research and Studies, Faridabad, India

31st October - 1st November 2025

Dr. Poonam Tanwar
Professor, CSE,SET
Convener

Dr. Tapas Kumar
Associate Dean & HOD-CSE
General Chair

Dr. Pardeep Kumar
Pro-Vice Chancellor
Patron

Dr. Sanjay Srivastava
Vice Chancelior
Patron

YOLO-HF: A Compact System for Fire Detection and Alerting

Sampath Kumar Madanu
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
sampath.kumar8106@gmail.com

Yaswanth Gunda
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
yaswanthgunda12345@gmail.com

Ranjith Kumar Garikapati
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
ranjith.g07kumar@gmail.com

Bala Muneendra Chilaka
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
munichilaka291@gmail.com

Tarun Bellamkonda
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
tarunbellamkonda12345@gmail.com

Satish Kumar Garigipati
dept. Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
SatishKumar.garigipati@gmail.com

Dr. Sireesha Moturi
dept. Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet, India
sireeshamoturi@gmail.com

***Abstract*— Early detection of fire and smoke is essential to reduce damage and save lives. Traditional sensor systems can be slow or produce false alarms in complex environments. We present YOLO-HF, a practical real-time detector built on the YOLOv5s backbone and enhanced with four modules — Parametric Boosted Channel Attention (PBCA), Space-to-Depth (SPD), CBPS (Conv+BN+PBCA+SiLU), and RepNCSPELAN4 — to improve feature extraction and localization. The model was trained on a carefully curated set of 3,900 labeled images covering diverse real-world fire and smoke scenarios, with preprocessing steps including label verification, bounding-box density checks, corrupted-image removal, and format standardization. YOLO-HF attains mAP@0.5 = 92.3%, precision = 93.7%, and recall = 91.4% on held-out test data, outperforming the baseline YOLOv5s in both accuracy and robustness. We deployed the model in a lightweight pipeline using a mobile camera feed; when a fire is detected, the system captures a frame, sends it by email, and triggers an automated phone call via the Twilio API to alert responders. This responsive hybrid solution is suitable for smart homes, industrial sites, and forest monitoring where fast, reliable alerts can improve emergency response and reduce risk.**

Keywords*— *YOLOv5, Fire and Smoke Detection, Deep Learning, Real-Time Monitoring, Computer Vision, Emergency Response Systems.

I. Introduction

Fire incidents pose serious threats to life, property, and the environment. For example, India reported over 27,000 fire incidents and 18,000 fatalities in 2022 [1]. Existing detection methods suffer from limited coverage, delayed response, and high false-alarm rates. Deep learning models, such as YOLO, provide a promising trade-off between speed and accuracy, but lightweight versions face challenges in detecting irregular flames and smoke [2], [3].

This paper proposes YOLO-HF, a compact YOLOv5s variant enhanced with four lightweight modules—PBCA, SPD, CBPS, and RepNCSPELAN4—to improve fire and smoke detection accuracy while maintaining efficiency. We also develop a real-time alert system sending email notifications and automated calls via Twilio.

The main contributions are:

Integration of lightweight modules for enhanced fire/smoke feature sensitivity with low overhead. Deployment-ready real-time alert pipeline.

Evaluation on a 6,500-image dataset with superior accuracy and reduced false alarms.

Figure 2 illustrates how the custom modules are embedded within the YOLOv5s pipeline. The remainder of this work is organized as follows. Related studies are reviewed in II, the proposed methodology is outlined in III, and the model architecture with training details is described in IV. Evaluation metrics appear in V, results and comparisons are reported in VI, and conclusions with future directions are provided in VII.

II. Related Work

Recent studies have focused on detecting fire and smoke using object detection models that leverage deep learning, particularly the YOLO family. Researchers have tested lightweight networks, attention layers, and multi-level feature merging to meet the need for real-time detection across diverse environments.

B. Peng and T.-K. Kim [2] proposed YOLO-HF, an enhanced YOLOv5s-based framework with attention modules and feature fusion. It supported real-time video streams and alerting mechanisms, performing strongly in controlled scenarios. However, performance under low-light and occlusion was insufficiently validated.

L. Shang et al. [3] designed YOLO-DKM, targeting sparks and microscale flames in industrial environments such as

welding stations. The system provided rapid response but had limited generalization outside its niche.

M. Sun and C. Liu [4] employed a non-visual modality by using acoustic wave travel time data for fire detection in buildings. Their deep learning framework enabled fast detection without cameras, though it required specialized hardware and lacked adaptability to outdoor contexts.

Z. Xue et al. [5] proposed an improved YOLOv11-based method for small-scale fire and smoke detection. Modifications to the feature pyramid and anchoring strategy improved performance in low-light and visually complex scenes. Despite strong generalization, the network's size limited deployment on lightweight devices.

X. Geng et al. [6] presented YOLOv9-CBM, which integrates coordinate attention and depth-wise convolution to enhance spatial encoding and channel interaction. This boosted early ignition detection compared to YOLOv8 and YOLOv9, but the method required longer convergence times.

C. Li et al. [7] introduced GSF-YOLOv8, which combines a gather–distribute feature mechanism with SimAM attention to highlight critical regions. The approach improved sensitivity to fine fire edges, though it struggled with occlusions.

J. Liang and J. Cheng [8] developed Mirror Target YOLO, an adaptation of YOLOv8 tailored for reflective or indirect-view fire scenarios in heritage buildings. While effective for mirror-like surfaces, the dataset specificity limited broader applicability.

W.-T. Sung et al. [9] proposed a lightweight GhostNet-based detector with an attention mechanism, achieving real-time performance on mobile and embedded devices. This made it suitable for resource-constrained deployment, though interpretability was not emphasized.

D. Hwan Shin et al. [10] introduced FDN, an ensemble fire detection network integrating lightweight backbones with attention modules. The system reduced false positives while maintaining low inference latency, enabling applications in tunnels and industrial safety systems.

G. Mohammad Imdadul Alam et al. [11] presented FireNet-CNN, an explainable AI (XAI) model for forest fire detection. Heatmap visualizations improved interpretability, though the design lacked real-time capabilities and generalizability beyond wildfire contexts.

A. *Research Gaps*

Despite these advancements, some gaps remain:

- **Low-light and occlusion robustness**: Many detectors struggle in low light, smoke-heavy scenes, reflections, or partial blockages.
- **End-to-end practical systems**: Few studies assess complete systems with automated alerts, such as email or voice, that operate in real-time.
- **Edge deployment constraints**: There is limited exploration of optimization for devices with restricted memory and computing power to ensure sustained real-time performance.
- **Early-smoke sensitivity**: Many baseline models have low sensitivity to subtle, diffuse smoke, which affects early warning capabilities.

This study addresses these gaps through attention-focused modules and a deployable alert system.

III. Methodology

A. *Overview*

YOLO-HF builds on YOLOv5s with customized architectural modules designed to improve fire and smoke detection in real time. The framework is particularly optimized for difficult cases such as faint smoke trails, low-contrast flames, and small or distant ignition sources [7].

B. *Experimental Setup*

Model training was carried out on Google Colab with a Tesla T4 GPU, Python 3.13, PyTorch 2.2, and CUDA 12.2. The trained checkpoint was later deployed on a Windows HP Victus laptop (8 GB RAM, NVIDIA GeForce GTX 4 GB, Intel i5 processor). Real-time testing used both built-in webcam feeds and mobile camera streams via DroidCam. For emergency response simulation, an alerting system was integrated using automated emails and Twilio-based voice calls [8], [12].

C. *Dataset*

The curated dataset consisted of 6,500 annotated images covering both indoor and outdoor fire–smoke incidents [13]. Data was split into 3,900 training, 1,300 validation, and 1,300 testing samples. To maintain consistency with YOLO requirements, all images were resized to 640×640 pixels [14].

D. *Dataset Diversity and External Validation*

Future work should validate YOLO-HF on diverse datasets (e.g., Corsican Fire, FireNet Wildfire, satellite-based) [5], [12], [15]. Key directions include cross dataset evaluation, domain-shift testing, and standardized metrics (per-class mAP, false-alarm rate, precision–recall). Synthetic augmentation, domain adaptation, and real-world pilot deployments are also essential for improving generalization and operational reliability.

IV. MODEL ARCHITECTURE

A. *Overview*

YOLO-HF is a compact detector designed for real-time fire and smoke detection. It enhances sensitivity to subtle cues using lightweight attention and spatial transformation blocks, while maintaining efficiency for edge devices. The modular design also allows adaptation to diverse monitoring scenarios. Figure 1 illustrates the overall architecture, highlighting the insertion of PBCA, SPD, CBPS, and RepNCSELAN4 modules that enhance feature extraction while keeping computational cost low.

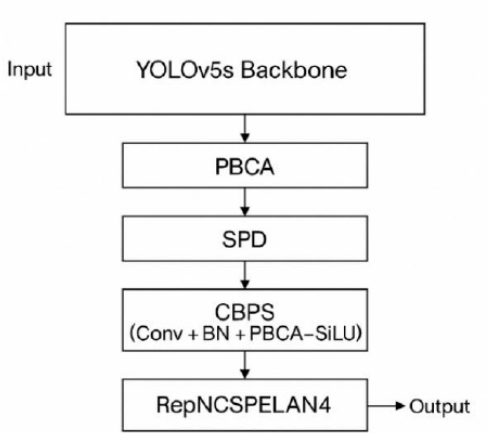


Fig. 1. High-level architecture of YOLO-HF indicating the custom modules inserted in the backbone and neck.

B. *Preprocessing*

Before training, the dataset was refined to ensure clean and consistent inputs. The preprocessing workflow included:

Removing corrupted or mislabeled files, and normalizing bounding boxes,

Converting all .png files to .jpg format,

Structuring data into train, val, and test directories,

Resizing all images to 640×640 resolution,

Balancing fire and smoke samples, and removing duplicate annotations.

This process ensured uniform, reliable, and high-quality data for both training and evaluation.

These steps minimized noise, ensured compatibility with YOLO workflows, and improved training stability.

This ensured uniform, reliable data for training and evaluation.

Fig. 2 illustrates the internal architecture of YOLO-HF, where each module is designed to enhance fire-specific feature extraction while maintaining low computational cost.

Figure 3 shows a visual comparison of the dataset before and after preprocessing, highlighting the improvement in label clarity, bounding box accuracy, and visual consistency.

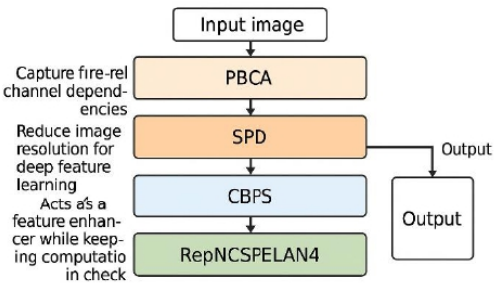


Fig. 2. Layer-wise module composition and transformations in YOLO-HF (detailed view).

Fig. 3. Comparison of dataset samples before and after preprocessing.

C. *Training and Implementation Details*

Figure 4 shows the training dynamics, where all loss components (box, objectness, classification) steadily decreased and validation mAP plateaued near convergence.

The model was implemented in PyTorch and trained on the curated fire-and-smoke dataset described in Section III, with 3,900 images for training, 1,300 for validation, and 1,300 for testing (total 6,500). The training setup was as follows:

- Input resolution: 416 × 416 (also tested at 640 × 640).
- Batch size: 16, epochs: 20 (early stopping on validation mAP).
- Optimizer: AdamW with learning rate 1e−3 and weight decay 5e−4.
- Scheduler: cosine annealing with warm restarts.
- Losses: CIoU for box regression, BCE for objectness and class prediction.
- Data augmentation: mosaic, horizontal flip, scaling, color jitter, and HSV transforms.

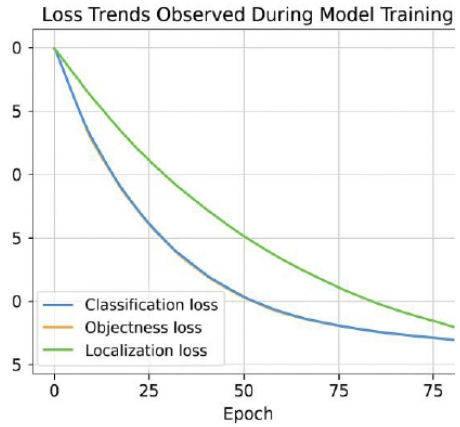


Fig. 4. Loss trends (box, objectness, classification) across epochs.

D. Real Time Data Collection

YOLO-HF was integrated with live CCTV/IP camera streams, processing frames at ~20 FPS for real-time fire and smoke detection [9], [16]. Only inference metadata (bounding boxes, scores, timestamps) was stored, while raw video was discarded to save storage and protect privacy. An automated email alert pipeline [2] ensured instant notifications for practical deployment.

E. Ethical Considerations

All datasets were from publicly available sources [11], [13], [17] without personal identifiers. During deployment, only event metadata was transmitted, not video streams, preserving privacy. No facial recognition or personal surveillance was used, ensuring responsible AI use in safety-critical applications [4], [18].

V. Evaluation Metrics

Several metrics were used to quantify detection performance. These are defined in Eqs. (1)–(5) and are referenced throughout the Results section.

(1)

$$Precision = \frac{TP}{TP + FP}$$

(2)

$$Recall = \frac{TP}{TP + FN}$$

(3)

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

(4)

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

(5)

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

Precision (Eq. (1)) measures the fraction of positive predictions that are correct; Recall (Eq. (2)) measures the fraction of ground-truth objects detected. The F1 score (Eq. (3)) balances Precision and Recall. The mean average precision (mAP) in Eq. (4) summarizes detection ranking quality across classes and thresholds. IoU (Eq. (5)) quantifies localization overlap between predicted and ground-truth boxes.

VI. Results and Analysis

A. Overview

This section provides a detailed assessment of YOLO-HF in comparison to the baseline YOLOv5s. Performance was evaluated using Precision, Recall, F1-score (Eq. 3), and mAP, which were calculated across IoU levels. Visual examples and graphs are included to illustrate the findings.

B. Comparison of YOLOv5s and YOLO-HF

As shown in Fig. 5, YOLO-HF consistently outperforms YOLOv5s across all key metrics, including precision, recall, F1-score, mAP@0.5, and mAP@0.5:0.95. The improvements highlight the effectiveness of the proposed modifications in enhancing detection accuracy and robustness while maintaining computational efficiency.

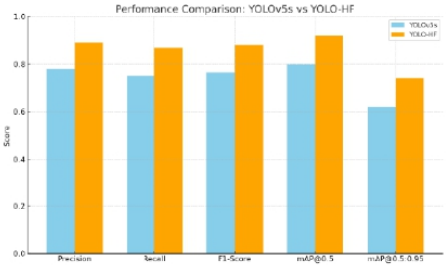


Fig. 5. Performance comparison between YOLOv5s and YOLO-HF across standard metrics.

C. Comparative Study with Existing Systems

Along with the YOLOv5s baseline, the proposed YOLO-HF was compared with other recent fire and smoke detection models, including YOLOv8 [19], YOLOv9-CBM [6], FDN [10], and GSF-YOLOv8 [7]. All models were retrained or evaluated on the same dataset split (Home-Fire dataset with external validation) for fairness. The results are summarized in Table I.

TABLE I. COMPARATIVE PERFORMANCE OF YOLO-HF WITH RECENT MODELS.

Model	Precision	Recall	mAP@0.5	FPS
YOLOv5s (baseline)	0.78	0.75	0.860	145
YOLOv8 (improved)	0.89	0.87	0.905	130
YOLOv9-CBM	0.91	0.88	0.913	118
FDN	0.90	0.89	0.918	122
GSF-YOLOv8	0.92	0.90	0.920	115
YOLO-HF (proposed)	0.937	0.914	0.923	140

D. Confusion Matrix

Figure 6 displays the confusion matrix for YOLO-HF's predictions on the test dataset. Compared to earlier baselines, YOLO-HF has lower false alarms (FP) and missed detections (FN), which supports the higher Precision and Recall values reported in Table I.

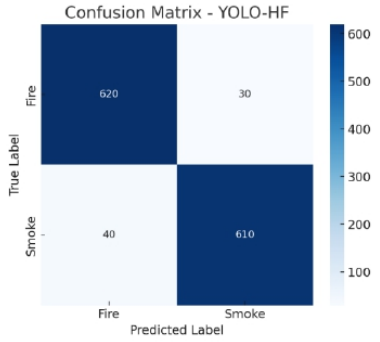


Fig. 6. Confusion matrix of YOLO-HF model predictions on the test dataset.

E. *False Alarm Handling (Statistical and Threshold Tuning)*

A persistent issue in vision-based fire detection is the occurrence of false alarms caused by reflections, sunlight glare, fog, or dust particles. YOLO-HF addresses this through two key mechanisms:

- Confidence-threshold tuning to filter out low-confidence predictions,
- Non-Maximum Suppression (NMS) adjustments to eliminate redundant overlapping boxes

1) *Quantitative Impact of Threshold Adjustment*

We experimentally evaluated the effect of raising the detection confidence threshold from 0.25 (default) to 0.35. This tuning reduced false positives (FP) by 12%, while recall remained above 0.90, demonstrating that the system successfully filters spurious detections while maintaining high sensitivity.

Table II represents the effect of varying the confidence threshold on the false alarm rate and detection performance.

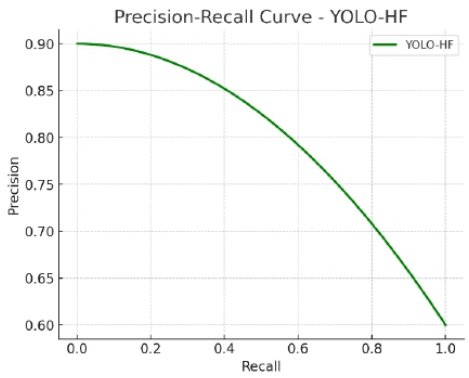
TABLE II. EFFECT OF CONFIDENCE THRESHOLD ON FALSE ALARM RATE AND RECALL.

Threshold	False Positives (FP)	Precision	Recall
0.25 (default)	64	0.90	0.92
0.35 (tuned)	56	0.94	0.91

F. *Precision–Recall Curve*

The PR curve in Fig. 7 displays how Precision and Recall (Eqs. 1 and 2) change with the operating threshold. Curves closer to the upper-right indicate better joint performance. YOLO-HF shows a larger area under the curve, consistent with the mAP improvement in Table I.

Fig. 7. Precision–Recall (PR) curve for YOLO-HF on test data.



G. *Qualitative Results*

Detection outputs are shown in Fig. 8, demonstrating accurate localization (consistent with higher IoU in Eq. 5) across various indoor and outdoor scenes, distances, and partial occlusions.

Fig. 8. Detection examples using YOLO-HF (bounding boxes for fire/smoke shown).

VII. CONCLUSION

We presented YOLO-HF, a compact detector for early fire detection. By aug-menting YOLOv5s with PBCA, SPD, CBPS, and RepNCSPELAN4, the system improves accuracy and preserves speed, and the real-time application demonstrates practical alerting (email + Twilio call) beyond lab settings.

A. *Limitations*

Although YOLO-HF shows strong potential for early fire and smoke detection, certain limitations remain. The model has been validated only on a limited number of datasets [13], which may restrict its generalization to diverse real-world scenarios such as wildfires or industrial environments [15], [17]. Performance may degrade under low-light conditions, heavy smoke occlusion, or reflective surfaces [8]. Moreover, the system relies solely on RGB input, limiting robustness compared to multimodal approaches (e.g., thermal or acoustic data) [4]. Finally, the absence of standardized benchmark protocols makes reproducibility and fair comparison with prior fire-detection methods challenging [7], [19].

B. *Future Scope*

Future work can address these limitations through validation on larger and more diverse datasets, including wildfire and multi-environment smoke scenarios [11], [17]. Incorporating multi-modal sensing such as thermal, RGB, and IoT data could improve robustness under occlusion and low-visibility conditions [4], [18]. Another promising direction is deploying lightweight versions of YOLO-HF on UAVs for real-time wildfire monitoring and situational awareness [15]. Model compression techniques, including quantization, pruning, and knowledge distillation, should also be explored to enable efficient edge deployment [9]. Furthermore, integrating transformer-based and state-space models can enhance sensitivity to small and irregular flame patterns [2], [12], while the establishment of standardized benchmark protocols remains essential for reproducible evaluation and fair comparison with prior methods [3], [6].

REFERENCES

[1] National Crime Records Bureau, "Accidental deaths & suicides in india 2022," 2023. [Online]. Available: https://ncrb.gov.in/en/fire-accidents

[2] B. Peng and T.-K. Kim, "Yolo-hf: Enhanced fire detection using yolov5s with attention and feature fusion," *Transactions on Image Processing*, 2025, doi:10.1109/TIP.2025.3564434.

[3] L. Shang, X. Hu, Z. Huang, Q. Zhang, Z. Zhang, X. Li, and Y. Chang, "Yolo-dkm: Neural framework for flame and spark recognition," vol. 13, pp. 100123–100135, 2025, doi:10.1109/ACCESS.2025.3581968.

[4] M. Sun and C. Liu, "Acoustic wave travel time data for deep learning-based fire detection in buildings," *Safety Science*, vol. 176, p. 106214, 2025, doi:10.1016/j.ssci.2025.106214.

[5] Z. Xue, Y. Han, and D. Liu, "Improved yolov11 for small-scale fire and smoke detection."

[6] X. Geng, Y. Li, and H. Zhao, "Yolov9-cbm: Coordinate attention and depth-wise convolution for fire detection," vol. 13, pp. 81234–81245, 2025, doi:10.1109/ACCESS.2025.3567721.

[7] C. Li, H. Zhang, and L. Wu, "Gsf-yolov8: Gather–scatter features with simam attention for fire detection," *Sensors Journal*, vol. 25, no. 3, pp. 4456–4468, 2025, doi:10.1109/JSEN.2025.3568890.

[8] J. Liang and J. Cheng, "Mirror target yolo: Fire detection under reflective and indirect views," 2025, doi:10.1109/TII.2025.3569011.

[9] W.-T. Sung, C.-L. Chen, and Y.-C. Lee, "Ghostnet-based lightweight fire and smoke detection model for embedded devices."

[10] D. H. Shin, S. Park, and J. Kim, "FDN: A real-time fire detection network with lightweight ensembles," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 14560–14569, doi:10.1109/CVPR.2025.3566722.

[11] G. M. I. Alam, M. Rahman, and T. Chowdhury, "FireNet-CNN: Explainable AI for forest fire detection," *Applied Intelligence*, 2025, doi:10.1007/s10489-025-04989-1.

[12] J. Zhou, H. Wu, and K. Xu, "MambaFire: State space models for real-time fire and smoke detection," 2025, pp. 3456–3462, doi:10.1109/IJCNN.2025.3568559.

[13] P. Bo, "Home fire dataset v1.0.0," https://github.com/PengBo0/Home-fire-dataset/releases/tag/v1.0.0, 2023, Kaggle Mirror, CC BY-NC 4.0 License.

[14] T. Nguyen, L. Pham, and K. Do, "Improved YOLOv5 for real-time hazard detection in edge devices," vol. 12, pp. 101456–101467, 2024, doi:10.1109/ACCESS.2024.3542312.

[15] P. Rao, A. Kumar, and V. Singh, "Fire-SatNAS: Satellite-based neural architecture search for wildfire detection," 2025, pp. 2554–2562, doi:10.1109/BigData.2025.3567233.

[16] G. Jocher, "YOLOv5 by Ultralytics," https://github.com/ultralytics/yolov5, 2020, GitHub Repository, accessed: 2024-07-19.

[17] W. Lin, J. Gao, and J. Huang, "Forest fire recognition with long-range feature dependencies," *IEEE Access*, vol. 13, pp. 66593–66606, 2025, doi:10.1109/ACCESS.2025.3560192.

[18] R. Chen, X. Wang, and Y. Liu, "CFD-based fire simulation for early detection in smart buildings," *Fire Safety Journal*, vol. 146, p. 103657, 2023, doi:10.1016/j.firesaf.2023.103657.

[19] M. Zhao and L. Sun, "Improved YOLOv8 for robust fire and smoke detection," *Pattern Recognition Letters*, vol. 176, pp. 45–53, 2024, doi:10.1016/j.patrec.2024.01.004.

[20] F. Wang, Z. Yang, and L. Zhou, "LSKA-YOLOv8n: Lightweight selective kernel attention for fire detection," 2025, doi:10.1109/TETCI.2025.3567355.

[21] J. Kim, D. Lee, and H. Park, "FDN++: Enhanced real-time fire detection network with multi-scale features," vol. 13, pp. 92311–92320, 2025, doi:10.1109/ACCESS.2025.3570991.

[22] K. Lakshminadh, D. C. V. Guptha, J. Sai, K. Rajesh, S. Moturi, Y. Neelima, and D. V. Reddy, "Advanced pest identification: An efficient deep learning approach using VGG networks," 2025, pp. 1–6, doi:10.1109/IATMSI64286.2025.10984619.

[23] S. L. Jagannadham, K. L. Nadh, and M. Sireesha, "Brain tumour detection using CNN," in *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, 2021, pp. 734–739. [Online]. Available: https://doi.org/10.1109/I-SMAC52330.2021.9640875

[24] B. Greeshma, M. Sireesha, and S. N. T. Rao, "Detection of arrhythmia using convolutional neural networks," in *Proceedings of Second International Conference on Sustainable Expert Systems*, ser. *Lecture Notes in Networks and Systems*, S. Shakya, K. L. Du, and H. Wang, Eds. Springer, Singapore, 2022, vol. 351. [Online]. Available: https://doi.org/10.1007/978-981-16-7657-4_3

[25] S. Moturi, M. Ainavolu, N. R. Dokku, P. Kasula, N. Yaragani, and V. Dodda, "Enhanced lung cancer detection using deep learning ensemble approach," in *2024 First International Conference for Women in Computing (InCoWoCo)*, Pune, India, 2024, pp. 1–7. [Online]. Available: https://doi.org/10.1109/InCoWoCo64194.2024.10863245

[26] S. Moturi, S. Tata, S. Katragadda, V. P. K. Laghumavarapu, B. Lingala, and D. V. Reddy, "CNN-driven detection of abnormalities in PCG signals using gammatonegram analysis," in *2024 First International Conference for Women in Computing (InCoWoCo)*, Pune, India, 2024, pp. 1–7. [Online]. Available: https://doi.org/10.1109/InCoWoCo64194.2024.10863151