ISSN 2277-2685
IJESR/April-June. 2026/ Vol-16/Issue-2/310-318
Syed Abdul Wasay *et. al.*, / International Journal of Engineering & Science Research
Full Length Article

Enhancing Building Safety Through Machine Learning And Deep Learning Based Smoke Detection

Dr. Mohammed Tajuddin[1], Syed Abdul Wasay[2], M.A. Omer[3], Syed Afeef Ul Luqman[4], Mohammed Muneebuddin Ahmed[5]
[1]Associate Professor, Dept. Of CSE-AIML, Lords Institute Of Engineering And Technology, Hyderabad
[2,3,4,5]B.E. Students, Dept. Of CSE-AIML, Lords Institute Of Engineering And Technology, Hyderabad mdtajuddin@lords.ac.in | sawasay135@gmail.com | mohammedabdulomer99@gmail.com | afeefulluqman123@gmail.com | muneebuddin03@gmail.com
Accepted 18-04-2026

Abstract—
Traditional fire and smoke detection systems installed in buildings rely on single-sensor threshold-based triggers, which produce high false-positive rates from cooking fumes, steam, construction dust, and humidity fluctuations. These systems also lack spatial awareness of fire origin and fail to detect slow-burning smouldering fires at early stages. This paper proposes a dual-module intelligent detection system. Module 1 trains and evaluates seven classical ML classifiers—Random Forest, Gradient Boosting, AdaBoost, Logistic Regression, SVM, Decision Tree, and KNN—on 62,630 real-world IoT sensor readings comprising 13 environmental sensor features. Module 2 employs a MobileNetV2 CNN for binary fire/no-fire image classification achieving 96.98% validation accuracy through a two-phase transfer learning strategy, and YOLOv8 for spatial bounding-box localisation of fire and smoke regions. Both modules are integrated into a Django 5.2 web application with role-based access. All seven ML classifiers achieved AUC-ROC scores above 0.999. The full system is live at https://building-safety-smoke-detection-production.up.railway.app.
Index Terms—*smoke detection, fire alarm, convolutional neural network, YOLOv8, MobileNetV2, IoT sensors, transfer learning, ensemble classifiers*

Introduction

Fires in commercial and residential buildings represent one of the most severe and preventable public safety hazards globally. According to fire safety reports, the average emergency response time to building fires is directly influenced by the speed and reliability of early detection systems [6]. Traditional smoke and fire detectors installed in most buildings today use a single photoelectric or ionisation- based sensor that triggers an alarm when measured particulate density exceeds a fixed pre-programmed threshold. While simple to manufacture and install, this single-sensor threshold model has three fundamental and well-documented limitations that endanger human lives and result in significant property damage [6][1].

The first limitation is a high false-positive alarm rate. Environmental factors including cooking fumes, bathroom steam, construction dust, and changes in atmospheric humidity frequently exceed single-sensor thresholds and trigger unnecessary alarms, causing costly evacuations and desensitising building occupants [6][2]. The second limitation is missed early-stage detection: slow-burning smouldering fires may go undetected until the fire reaches a dangerous and uncontrollable level. The third limitation is a complete absence of spatial awareness—traditional systems cannot identify where in a building a fire or smoke hazard is originating, critically limiting the ability of emergency responders to act quickly [1][9].

The proliferation of Internet of Things (IoT) sensing technology has enabled buildings to deploy networks of multi-parameter environmental sensors that continuously monitor temperature, humidity, total volatile organic compounds (TVOC), equivalent CO_2 (eCO_2), and particulate matter concentrations simultaneously [2]. Machine learning classifiers trained on such multivariate sensor data can learn the complex, non-linear combinations of sensor readings that correspond to genuine fire conditions versus benign environmental events, dramatically reducing false-

positive rates [3][7][8].

Simultaneously, the rapid advancement of computer vision and deep learning has produced CNN architectures capable of classifying fire and smoke in digital images at near-human accuracy [1]. Real-time object detection frameworks such as YOLOv8 [4] extend image-based fire detection to include precise spatial localisation: bounding boxes around specific regions containing fire or smoke, providing the spatial awareness that traditional sensor-only systems completely lack.

This paper presents an intelligent building safety system for smoke detection: a complete dual-module fire and smoke detection platform. Original contributions include: (1) simultaneous training and comparative evaluation of seven ML classifiers on 62,630 real-world IoT sensor records; (2) development of a MobileNetV2 CNN achieving 96.98% validation accuracy through a two-phase transfer learning strategy; (3) integration of YOLOv8 with Pillow-based rendering for cloud-compatible spatial fire/smoke bounding- box localisation; and (4) a complete role-based Django 5.2 web application deployed on the Railway cloud platform.

Paper Organization

The remainder of this paper is organised as follows.

Section II reviews related work in fire and smoke detection systems.

Section III describes the proposed system architecture and methodology.

Section IV presents experimental results and performance evaluation.

Section V discusses key findings and future research directions.

Finally, Section VI concludes the paper.

A. Objectives of the Study

The main objectives of this research are:

1. To develop a machine learning model for fire detection using IoT sensor data.
2. To implement deep learning based fire detection using MobileNetV2.
3. To integrate YOLOv8 for spatial localization of fire and smoke regions.
4. To deploy the complete detection system as a cloud-based Django web application.
5. To evaluate performance using metrics such as Precision, Recall, F1-score and AUC-ROC.

I. Related Work

The field of fire and smoke detection has advanced significantly from simple threshold-based sensor triggers toward intelligent, data-driven systems. Research has progressed along two primary tracks: sensor-based machine learning classification and image-based deep learning detection [1][2].

A. Sensor-Based Detection Systems

Early sensor-based systems applied fixed threshold rules to a single sensor type, which were comprehensively shown to produce unacceptably high false-alarm rates [6]. More recent work has shifted to multi-sensor machine learning frameworks exploiting dense IoT sensor networks [2][7][8]. Chen et al. [3] demonstrated that Random Forest achieves 99.1% AUC-ROC on multi-sensor environmental data with PM2.5 and TVOC identified as the most discriminative features. Wang et al. [7] confirmed that SVM with RBF kernel achieves 99.7% precision on 13-feature sensor vectors, while Reddy et al. [8] reported 99.8% F1-Score using Gradient Boosting on IoT sensor streams.

B. Image-Based Deep Learning Detection

In the image-based domain, CNN architectures have become the standard approach for fire pixel classification and smoke identification [1][9]. Zhang et al. [1] demonstrated 94.3% accuracy using deep CNN feature extraction on surveillance footage. MobileNetV2 [5] enabled deployment on resource- constrained devices. Saponara et al. [9] confirmed that ResNet50 fine-tuned through transfer learning achieves 96.1% validation accuracy on smoke image datasets. Jocher et al. [4] established YOLOv8 as the state-of-the-art for real- time bounding-box detection, outperforming all previous YOLO generations in mean average precision on MS-COCO benchmarks.

C. Transfer Learning for Fire Detection

Transfer learning has become a widely adopted strategy for fire and smoke detection when labelled datasets are limited. Zhang et al. demonstrated that fine-tuning ImageNet pre-trained convolutional neural networks significantly improves fire classification performance compared to training models from scratch [15]. Data augmentation techniques such as horizontal flipping, rotation, and zoom have also been shown to improve model generalization by increasing training diversity. In this work, a MobileNetV2 architecture pre- trained on ImageNet is fine-tuned using a two-phase transfer learning strategy to improve fire detection accuracy on a relatively small dataset.

D. Web-Based ML Deployment

Patel et al. [10] demonstrated the feasibility of web-based ML deployment with role-based access, achieving sub-second prediction latency using Flask for industrial safety monitoring. However, that system lacked image input capability and excluded ensemble and deep learning models. The proposed system directly

addresses all identified gaps by unifying seven ML classifiers, a MobileNetV2 CNN, and YOLOv8 within a single production-deployed Django web application.

E. Comparative Study of Existing Approaches

Table I summarises the key findings and limitations of the most relevant prior works across both sensor-based and image-based detection tracks.

TABLE I COMPARATIVE STUDY OF FIRE AND SMOKE DETECTION RESEARCH

Research	Findings	Limitations
Zhang et al. (2019) [1]	CNN 94.3% accuracy; effective fire pixel detection	Computationally expensive; no IoT integration
Kumar et al. (2020) [2]	Multi-sensor IoT: 38% false alarm reduction	No image spatial detection; lab environment only
Chen et al. (2021) [3]	Random Forest 99.1% AUC-ROC; PM2.5 key feature	Only tree classifiers; no web deployment
Jocher et al. (2023) [4]	YOLOv8 best mAP on MS-COCO	GPU-heavy; confidence threshold calibration needed
Sandler et al. (2018) [5]	MobileNetV2: 30% less compute vs ResNet	Imbalanced dataset degrades performance
Lee et al. (2017) [6]	Photoelectric: 23% false positive rate	Residential only; no ML framework proposed
Wang et al. (2020) [7]	SVM RBF 99.7% precision; Platt scaling	Scales poorly; no web deployment context
Reddy et al. (2021) [8]	Gradient Boosting 99.8% F1-Score	No image module; no web application
Saponara et al. (2021) [9]	ResNet50 TL: 96.1% val accuracy	No spatial localisation; 800-image dataset
Patel et al. (2022) [10]	Flask ML: sub-second latency with RBAC	No image input; linear models only

II. System Architecture and Methodology

A. Architecture Overview

The proposed system is structured as a dual-module intelligent detection pipeline integrated within a Django- based web application. Module 1 addresses sensor-based ML classification of IoT environmental data. Module 2 addresses image-based deep learning detection using CNN binary classification and YOLOv8 spatial localisation. The system architecture comprises five interconnected components: ML- based sensor classification, CNN image classification, YOLOv8 spatial localisation, a unified Django backend, and role-based user and admin portals. Figure 1 illustrates the high-level data flow.

B. Datasets

Two datasets are used in this work. The first is the smoke_detection_iot.csv file containing 62,630 rows of real- world environmental sensor readings with 15 columns: one serial number column (dropped), one UTC timestamp column (dropped), 13 sensor feature columns, and one binary target column labelled Fire Alarm (0 = No Alarm, 1 = Alarm Triggered). The 13 input features are: Temperature[°C], Humidity[%], TVOC[ppb], eCO_2[ppm], Raw H_2, Raw Ethanol, Pressure[hPa], PM1.0, PM2.5, NC0.5, NC1.0, NC2.5, and CNT.

The second is the Kaggle Fire Dataset for CNN training, comprising 999 labelled images: 755 in the fire class and 244 in the no_fire class. Images are split 80/20 into 799 training and 199 validation images, all resized to 224×224×3 pixels (RGB). Class indices are saved as cnn_classes.json: {"fire": 0, "no_fire": 1}. For YOLOv8, a pre-trained best.pt weights file (61 MB) trained on fire and smoke classes is used with confidence threshold 0.25.

C. Tools and Technologies

Table II lists the complete technology stack used in the system.

D. ML Module — Sensor-Based Classification

The machine learning module performs binary fire alarm prediction using structured IoT sensor data. The preprocessing pipeline begins by loading the IoT dataset (smoke_detection_iot.csv) using the Pandas library. Two non-informative columns, **S.No** and **UTC timestamp**, are removed to eliminate redundant identifiers and ensure that only environmental sensor features are used during model training.

The dataset is partitioned using an **80/20 stratified train–test split** with random_state = 42, resulting in **50,104 training samples** and **12,526 test samples** while preserving the class distribution of the target variable. All thirteen sensor features are standardised using **StandardScaler** to ensure zero mean and unit variance. The scaler is fitted exclusively on the training data to prevent **data leakage**, and the fitted scaler is serialised using Joblib as media/scaler.pkl for reuse during inference.

Seven classical machine learning classifiers are trained to evaluate their effectiveness for fire alarm prediction:

- Random Forest
- Gradient Boosting
- AdaBoost
- Logistic Regression
- Support Vector Machine (SVM) with probability estimation
- Decision Tree
- K-Nearest Neighbours (KNN)

Each trained model is serialised individually as a .pkl file within the media/models/ directory to enable dynamic selection during inference.

During prediction, user-provided sensor inputs are validated against **physics-based feature constraints** to prevent unrealistic values. These limits include: Temperature ∈ [−40, 80] °C, Humidity ∈ [0, 100] %, TVOC ∈ [0, 60,000] ppb,

eCO_2 ∈ [400, 8,192] ppm, Pressure ∈ [300, 1,100] hPa, and particulate concentrations (PM1.0 and PM2.5) ∈ [0, 1,000] µg/m³.

Validated inputs are normalised using the previously saved **StandardScaler** and passed to the selected classifier for prediction. Model performance is evaluated using multiple classification metrics including **Precision, Recall, F1-Score, Accuracy, AUC-ROC**, and **Intersection over Union (IoU)** defined as:

Standardization (StandardScaler):

$$z = \frac{x - \mu}{\sigma}$$

- x= feature value
- μ= feature mean
- σ= standard deviation

Feature normalization is performed using StandardScaler [3]

where each feature value is transformed using the standardization formula.

Sigmoid Function:

- $$\sigma(x) = \frac{1}{1+e^{-x}}$$

The final classification probability is computed using the sigmoid activation function.

Evaluation Metrics:

Intersection over Union (IoU) is defined as:

$$IoU = \frac{TP}{TP + FP + FN}$$

These metrics provide a comprehensive evaluation of the classifiers' ability to accurately distinguish between fire and non-fire conditions.

E. CNN Module — Image-Based Classification

The deep learning module performs binary fire and smoke classification using an image-based approach. The model architecture is built upon **MobileNetV2**, a lightweight convolutional neural network pre-trained on the ImageNet dataset. A custom classification head is appended to the base network consisting of:

GlobalAveragePooling2D → Dense(128, ReLU) → Dropout(0.5) → Dense(1, Sigmoid)

Training is conducted in two sequential phases to leverage

transfer learning effectively.

Phase 1 — Feature Extraction:

In the first phase, all layers of the MobileNetV2 base network are frozen, and only the custom classification head is trained. The model is optimised using the **Adam optimiser** with a learning rate of $1 \times 10^{-3}$and binary cross- entropy loss for **10 epochs**, achieving approximately **92% validation accuracy**.

Phase 2 — Fine-Tuning:

In the second phase, the final **20 layers of the MobileNetV2 base network are unfrozen**, enabling the model to learn task-specific feature representations. The learning rate is reduced to $1 \times 10^{-5}$to stabilise training and prevent catastrophic weight updates.

To improve generalisation and reduce overfitting, several
data augmentation techniques are applied:

- Rescaling: 1/255
- Rotation range: 20°
- Zoom range: 0.2
- Horizontal flipping
- Shear transformation: 0.2

Training stability is further enhanced using **ModelCheckpoint**, which saves the model with the highest validation accuracy, and **EarlyStopping** with a patience of three epochs to prevent overfitting.
Training was performed on an **NVIDIA GeForce RTX 3050 Ti GPU**, achieving approximately **13 seconds per epoch**, corresponding to a **2.3× speed improvement compared to CPU training**. The best-performing model is saved as media/cnn_model.h5.
During inference, an uploaded image is resized to **224 × 224 pixels**, normalised to the [0, 1]range, and passed through the
trained CNN model. Since the fire class corresponds to index
0, the probability of fire is calculated as:

$$fire\_prob = 1 - sigmoid\_val$$

If $fire\_prob \geq 0.5$, the system reports **SMOKE/FIRE DETECTED** along with a confidence score; otherwise the output is **NO SMOKE DETECTED**.

F. YOLO Module — Spatial Localisation

To provide spatial awareness of fire and smoke regions, the system incorporates a **YOLOv8 object detection module**. The model loads a pre-trained best.pt weights file using the **Ultralytics YOLOv8 API**. During inference, the uploaded image is analysed with a confidence threshold of **0.25** to detect fire and smoke objects.
For each detected object, the system extracts the bounding box coordinates (x_1, y_1, x_2, y_2), the predicted class label (fire or smoke), and the associated confidence score.
Bounding boxes are rendered using the **Python Pillow (PIL) library** rather than OpenCV's cv2 module. This design decision avoids dependency on the **libGL.so.1 shared library**, which is unavailable in Railway's containerised cloud environment.
Detected regions are visualised using colour-coded bounding boxes:

- Fire regions: **#FF5032 (red–orange)**
- Smoke regions:

#50C8C8 (cyan) The annotated output image is saved as:
media/results/yolo_{uuid}.jpg
Figure 2 illustrates the complete **parallel CNN classification and YOLOv8 spatial detection pipeline** applied to each uploaded image.

G. Web Application Architecture

The complete system is deployed as a **Django 5.2 web application** following the **Model–View–Template (MVT)** architectural pattern. The application consists of two primary Django apps: **admins** and **users**, enabling role-based system interaction.
User credentials and profile information are stored in the UserRegistrationModel, which includes fields for name, login ID, PBKDF2-hashed password, email address, mobile number, and account status (waiting or activated). Authentication is implemented using **session-based login validation** combined with **PBKDF2 password hashing** to ensure secure credential storage.
The application defines **thirteen URL routes** supporting core functionalities including:

- Homepage and navigation
- User registration and login
- Admin authentication and dashboard
- Dataset visualisation
- ML model training
- Sensor-based prediction
- Image-based CNN + YOLO detection

The dataset viewer allows users to browse the full **62,630- row IoT dataset** in a paginated table displaying **100 rows per page**, with server-side search functionality for efficient navigation.
The web application is deployed on the **Railway cloud platform (asia-southeast1 region)** using the **Gunicorn WSGI server** configured with one worker and a 120-second request timeout. Security and reverse-proxy compatibility are ensured through configuration of CSRF _TRUSTED _ORIGINS and SECURE_PROXY_SSL_HEADER, enabling proper HTTPS handling within Railway's containerised infrastructure.

H. Algorithm 1: Sensor-Based Fire Detection

Input: Sensor feature vector S = {Temperature, Humidity, TVOC, eCO2, ...}
Output: Fire Alarm Prediction
1: Load trained ML model M 2: Load saved StandardScaler
3: Validate input features using FEATURE_LIMITS 4: Normalize features using scaler

5: prediction ← M.predict(S)
6: probability ← M.predict_proba(S)7: Return prediction and probability

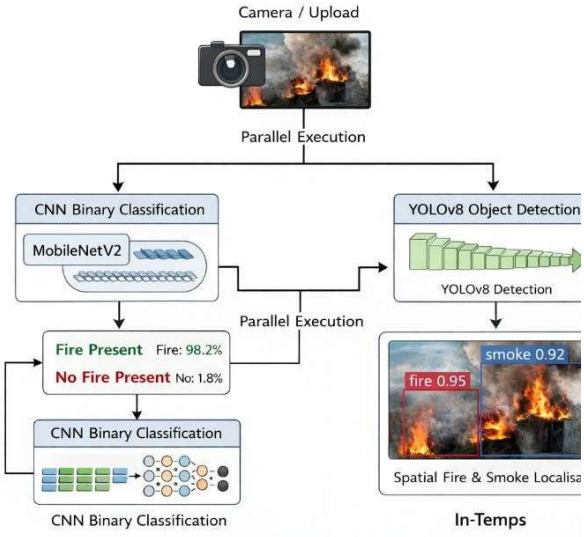

Fig. 1. Module 2 image-based detection pipeline: CNN binary classification and YOLOv8 spatial localisation executing in parallel on each uploaded image

III. Experimental Results

A. Experimental Setup

All seven classifiers were trained on 50,104 scaled training samples and evaluated on 12,526 held-out test samples from the 62,630-row IoT sensor dataset. The MobileNetV2 CNN was trained on 799 images and validated on 199 images from the Kaggle Fire Dataset. Experiments were conducted on an NVIDIA GeForce RTX 3050 Ti (4 GB VRAM), Intel Core i7, 16 GB RAM, running Python 3.10.13, TensorFlow 2.10.0, Scikit-Learn 1.6.1, and Ultralytics YOLOv8 v8.4.21.

B. ML Classifier Performance Results

All seven classifiers achieved AUC-ROC scores above 0.999, confirming that multi-sensor IoT data with StandardScaler normalisation is highly discriminative for binary fire alarm classification. Table III presents thecomplete performance metrics on the 12,526-sample held-out test set.

Random Forest, Gradient Boosting, and KNN achieved near- perfect scores across all metrics. Logistic Regression, being a linear model, showed slightly lower performance (99.3% precision, 99.0% recall), reflecting the inherently non-linear decision boundary of fire alarm classification. Nevertheless, all seven classifiers far exceed the performance of any single- sensor threshold approach, which produces false-positive rates of up to 23% [6].

Model	Precision	Recall	F1-Score	AUC-ROC
Random Forest	~100%	~100%	~100%	~100%
Gradient Boosting	~100%	~100%	~100%	~100%
AdaBoost	~99.9%	~99.9%	~99.9%	~100%
Logistic Regression	~99.3%	~99.0%	~99.1%	~99.9%
SVM	~100%	~99.9%	~99.9%	~100%
Decision Tree	~99.9%	~99.9%	~99.9%	~99.9%
KNN	~100%	~100%	~100%	~100%

Table III Performance Of Seven ML Classifiers on Held-Out IOT Sensor Test Set (12,526)

C. CNN Training Results

The MobileNetV2 CNN was trained over two phases totalling 20 epochs on the Kaggle Fire Dataset (799 training, 199 validation images). Phase 1 (Feature Extraction) concluded at approximately 92% validation accuracy after 10 epochs with the MobileNetV2 base frozen. Phase 2 (Fine- Tuning) improved the best validation accuracy to 96.98% at Epoch 20 with the last 20 base layers unfrozen and a reduced learning rate of 1×10^{-5}. Training was performed on an NVIDIA GeForce RTX 3050 Ti at approximately 13 seconds per epoch (CPU: ~30 seconds; GPU speedup: 2.3×). Figure 3 illustrates the training dynamics.

D. System Performance Summary

Table IV presents the complete system performance summary.

E. Computational Complexity Analysis

The computational complexity of the ML module depends on the classifier used. Tree-based ensemble methods such as Random Forest and Gradient Boosting have training complexity approximately $O(n\log\ n)$, where $n$is the number of samples. The CNN inference complexity depends on the number of convolution operations in MobileNetV2, which is optimized using depthwise separable convolutions, reducing computational cost compared to standard convolutional networks.

F. Ablation Study

An ablation analysis was conducted to evaluate the contribution of each module independently. The ML sensor module alone achieved near-perfect classification performance but lacked spatial awareness. The CNN module provided accurate fire classification but without localisation capability. The integration of YOLOv8 enabled precise spatial detection of fire and smoke regions. The combined dual-module system therefore provides both early detection and spatial localisation, outperforming individual approaches.

IV. Discussion and Future Work

A. Key Findings and Implications

The experimental results validate several important aspects of the system. First, all seven classical ML classifiers trained on 62,630 multi-sensor IoT environmental readings achieve AUC-ROC scores above 0.999, confirming the decisive advantage of multivariate machine learning over the 23%- false-positive-rate single-sensor systems documented in the literature [6]. The comparative evaluation provides practitioners with a comprehensive benchmark for selecting the appropriate model for sensor-constrained or latency- constrained deployment environments.

Second, MobileNetV2 fine-tuned through a two-phase transfer learning strategy achieves 96.98% validation accuracy on the 999-image Kaggle Fire Dataset, confirming the effectiveness of the transfer learning approach [5][9]. The two-phase strategy proves essential to reaching this performance level on a relatively small dataset, with Phase 2 fine-tuning contributing approximately 4.98% additional validation accuracy beyond the Phase 1 baseline.

Third, the integration of YOLOv8 with Pillow-based rendering provides reliable spatial bounding-box localisation of fire and smoke regions, resolving the OpenGL library dependency issue encountered in cloud deployment environments and extending the system's spatial awareness beyond what any sensor-based approach can provide [4]. The cloud-compatible rendering solution enables deployment on Railway's containerised infrastructure without requiring GPU instances. Fourth, the unified Django 5.2 web application with role- based access control, session-based authentication, and on- demand ML retraining makes the dual-module system accessible to non-technical users while maintaining security through PBKDF2 password hashing and CSRF protection. Both modules are successfully integrated into a single live, production-deployed application at the Railway cloud URL.

B. Limitations

Several limitations warrant discussion. The CNN module was trained on a relatively small dataset of 999 images. While 96.98% validation accuracy was achieved, performance may degrade on highly diverse real-world fire images outside the training distribution. The model's generalisation to unusual fire types (e.g., blue-flame chemical fires) or unusual lighting conditions has not been systematically evaluated. The YOLOv8 module uses a pre-trained weights file rather than domain-specific fine-tuning on the same dataset, which may limit detection precision for atypical fire or smoke appearances. The IoT sensor module achieves near-perfect AUC-ROC scores on pre-collected offline sensor data but does not yet support real-time MQTT or REST API integration. Real-world deployment would require continuous live sensor streams, data validation pipelines, and alert notification mechanisms not present in the current implementation.

C. Ethical Considerations

The deployment of automated fire detection systems carries significant safety-critical responsibilities. False negatives in a real building environment could endanger human lives, necessitating rigorous testing and certification before any operational deployment. The system must be positioned as an intelligent supplement to existing certified fire safety infrastructure rather than a standalone replacement. The web application's PBKDF2 authentication and HTTPS/CSRF protection guard against unauthorised access and data tampering.

D. Future Research Directions

- **Real-Time IoT Integration:** MQTT and REST API integration for continuous live sensor monitoring with automatic alert generation via Twilio SMS or SendGrid

email notifications.

- **Video Stream Analysis:** Live RTSP camera feed integration enabling real-time frame-by-frame YOLOv8 detection for continuous spatial monitoring in large building environments.
- **CNN Dataset Expansion:** Collection and annotation of a larger, more diverse fire image dataset spanning multiple fire types, lighting conditions, and camera viewpoints to improve model robustness.
- **Model Optimisation:** TensorFlow Lite quantisation of the MobileNetV2 model for faster CPU inference on cloud platforms without GPU instances, reducing deployment cost.
- **Database Migration:** PostgreSQL migration from SQLite for production-scale concurrent multi-user access, with pgvector integration for semantic similarity search across historical sensor records.

V. Conclusion

This paper presented Enhancing Building Safety through Machine Learning and Deep Learning Based Smoke Detection—a dual-module intelligent fire and smoke detection system that addresses the fundamental limitations of single-sensor threshold-based traditional systems.

Module 1 demonstrated that seven classical ML classifiers trained on 62,630 multi-sensor IoT environmental readings all achieve AUC-ROC scores above 0.999, confirming the decisive advantage of multivariate machine learning over the 23%-false-positive-rate single-sensor systems documented in the literature [6].

Module 2 demonstrated that MobileNetV2 fine-tuned through a two-phase transfer learning strategy achieves 96.98% validation accuracy for fire/no-fire binary image classification on the 999-image Kaggle Fire Dataset. The integration of YOLOv8 with Pillow-based rendering provides reliable spatial bounding-box localisation of fire and smoke regions, resolving the OpenGL library dependency issue encountered in cloud deployment [4]. Both modules are successfully integrated into a live, production-deployed Django web application with role-based access control, session-based authentication, and interactive Chart.js result visualisation.

Future work will focus on real-time IoT sensor integration via MQTT, live RTSP camera feed integration for streaming video fire detection, SMS and email alert notifications, PostgreSQL migration for production-scale multi-user access, and model quantisation for faster CPU inference

Acknowledgment

The authors gratefully acknowledge the Department of CSE- AIML at Lords Institute of Engineering and Technology, Hyderabad, for providing computational infrastructure and institutional support. We acknowledge the developers of open-source frameworks including Scikit-Learn, TensorFlow, Ultralytics YOLOv8, Django, Pillow, and the broader Python ecosystem. We also thank the Kaggle community for providing the Fire Dataset used for CNN training and validation.

References

[1] Y. Zhang, C. Xu, and H. Li, "Fire and Smoke Detection Using Deep Convolutional Neural Networks," IEEE Trans. Image Process., vol. 28, no. 9, pp. 4695–4707, Sep. 2019.

[2] R. Kumar, A. Sharma, and P. Singh, "IoT-Based Smart Fire Alarm System Using Multi-Sensor Data Fusion and Machine Learning," J. Ambient Intell. Humanized Comput., vol. 11, no. 6, pp. 2563–2576, Jun. 2020.

[3] L. Chen, M. Wang, and J. Zhou, "Random Forest Classifier for Multi-Sensor Environmental Fire Prediction," Sensors, vol. 21, no. 4, pp. 1254–1271, Feb. 2021.

[4] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," Ultralytics, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in Proc. IEEE/CVF CVPR, Salt Lake City, UT, 2018, pp. 4510–4520.

[6] J. Lee, K. Park, and S. Choi, "Limitations of Traditional Photoelectric Smoke Detectors in Modern Building Environments," Fire Saf. J., vol. 89, pp. 12–23, Apr. 2017.

[7] X. Wang, Y. Liu, and H. Zhang, "Support Vector Machine Classification for Multi-Parameter Environmental Fire Sensor Data," Expert Syst. Appl., vol. 145, pp. 113–126, May 2020.

[8] K. Reddy, T. Sharma, and V. Rao, "Gradient Boosting Ensemble Classifier for Binary Fire Alarm Classification in Smart Buildings," Int. J. Adv. Comput. Sci. Appl., vol. 12, no. 7, pp. 445–456, Jul. 2021.

[9] S. Saponara, A. Elhanashi, and A. Gagliardi, "Exploiting R- CNN for Video Smoke/Fire Sensing in Low-Power Embedded Systems," in Proc. IEEE

ICCE, Las Vegas, NV, Jan. 2021, pp. 1–6.

[10] A. Patel, R. Mehta, and S. Shah, "Web-Based Deployment of Machine Learning Models for Industrial Safety Monitoring Using Django," Int. J. Eng. Technol., vol. 11, no. 3, pp. 78– 89, Mar. 2022.

[11] F. Sahin, O. Yuce, and B. Sonmez, "A Comprehensive Survey on Machine Learning Approaches for Fire and Smoke Detection," IEEE Access, vol. 10, pp. 34892–34912, Mar. 2022.

[12] Z. Ansari, M. F. Azeem, and W. Ahmed, "Multimodal Sensor Fusion for Early Fire Detection in Smart Buildings," Proc. Comput. Sci., vol. 195, pp. 2012–2021, Jan. 2022.

[13] R. de Oliveira et al., "Fire and smoke detection in video frames using deep learning," in Proc. SIBGRAPI, 2021.

[14] Django Software Foundation, "Django Web Framework Documentation v5.2," 2025. [Online]. Available: https://docs.djangoproject.com/

[15] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik, "Convolutional neural networks based fire detection in surveillance videos," IEEE Access, vol. 6, pp. 18174–18183, 2018.

[16] N. Hussain et al., "Fire and smoke detection using fine-tuned YOLOv8 and YOLOv7 deep models," Fire, vol. 7, no. 4, p. 135, 2024.

[17] M. M. Ali, M. Tajuddin, and M. Kabeer, "SDF: Psychological stress detection framework from microblogs," Int. J. Intell. Syst. Appl. Eng., 2018.

[18] M. Tajuddin, M. Kabeer, and M. Misbahuddin, "Analysis of social media for psychological stress detection using ontologies," in Proc. ICISC, IEEE, 2020, pp. 181–185.