International Journal of Scientific Research and Modern Technology (IJSRMT) ijsrmt.com
Volume 4, Issue 7, 2025
DOI: https://doi.org/10.38124/ijsrmt.v4i7.650

Real-Time Video-Based Fire Detection Using Deep Learning Techniques: A Study of YOLO and CNN Architectures

Dr. Harish S Gujjar[1]

[1]Associate Professor and Head, Department of Computer Science, SSAS Government First Grade College, Hosapete, Karnataka, India.

Publication Date 2025/07/25

Abstract
Early and accurate fire detection is vital for preventing severe loss of life and property. Traditional fire detection systems based on sensors often suffer from delays and limited coverage. With advancements in computer vision and deep learning, video-based fire detection has become a promising alternative. This paper explores real-time fire detection in video streams using deep learning models, particularly YOLO (You Only Look Once) and Convolution Neural Networks (CNNs). We present a comprehensive methodology, covering dataset preparation, preprocessing, model training, and evaluation. The strengths and limitations of each approach are discussed, and experimental results demonstrate the effectiveness of YOLO and CNNs for timely and reliable fire detection.

***Keywords** - YOLOv5, CNNs, PyTorch, TensorFlow, OpenCV*

I. INTRODUCTION

Fires are catastrophic events that demand immediate attention. Traditional fire detection methods, including smoke detectors and temperature sensors, are often reactive and suffer from false alarms or detection delays. Video-based fire detection systems offer the advantage of wider coverage, faster response times, and the ability to monitor large open spaces.

The development of deep learning techniques, especially object detection frameworks like YOLO and CNN-based classifiers, has revolutionized automated fire detection. This research aims to design, implement, and evaluate a deep learning-based fire detection system that can efficiently analyze video streams and detect fire in real time.

Previous studies have explored fire detection through feature extraction methods based on color, motion, and shape analysis. However, these traditional techniques are often prone to false positives due to environmental factors like sunlight, headlights, or bright objects.

Deep learning models have shown remarkable improvements in object recognition tasks and are now being applied to fire detection:

- *CNNs* can automatically learn relevant features from fire images without manual feature engineering.
- *YOLO* models offer real-time object detection with high accuracy, making them suitable for video-based applications.

Researchers have also integrated techniques like LSTM (Long Short-Term Memory) networks for temporal information processing, though at the cost of increased computational complexity.

II. LITERATURE REVIEW

Fire detection has evolved significantly with the advancement of computer vision and artificial intelligence (AI). Traditional sensor-based methods, though effective in some cases, often suffer from high false alarm rates and limited range. Recent research emphasizes the importance of integrating vision-based and deep learning techniques for improved accuracy and efficiency.

Zhang and Wang (2025) proposed a vision-based fire detection approach that relies on smart surveillance systems to detect fire using visual features like shape, color, and motion. Their system significantly reduces false positives by leveraging multiple feature fusion techniques [1]. Earlier work by Singh and Chen (2022) also explored similar visual characteristics in fire detection, emphasizing the reduction of false alarms using computer vision algorithms [10].

With the advancement of deep learning, models like YOLO (You Only Look Once) have become prominent. Chen and Li (2025) introduced an enhanced YOLOv5s-RBC fire detection algorithm that integrates RepVGG to improve the model's lightweight characteristics without compromising accuracy [2]. Likewise, Lv (2024) employed YOLOv8n in fire and smoke detection, improving precision in smart factory environments with real-time capabilities [3].

Datasets play a crucial role in training reliable models. Smith and Doe (2024) developed a comprehensive open flame and smoke detection dataset tailored for deep learning research, addressing the lack of diverse and annotated datasets in the domain [4]. Johnson and Lee (2024) provided a broader survey on forest fire detection and prediction techniques, covering both traditional and modern AI-based methods [5].

For real-time embedded applications, Kumar and Patel (2023) optimized fire detection algorithms to run on resource-constrained devices using MobileNetV3-large, replacing heavier backbone networks like CSPDarkNet53, thus making deployment feasible in portable systems [6]. In indoor surveillance settings, Garcia and Nguyen (2023) proposed an early fire detection model using CCTV footage and deep learning, achieving effective results in enclosed environments like buildings [7].

Ahmed and Zhao (2023) employed deep learning-based models for detecting both fire and smoke in forest imagery. Their method combines AI with environmental data, proving effective in reducing response time for firefighting efforts [8]. Complementing this, Oluwaseun and Mbatha (2023) conducted a comparative survey of traditional sensor-based and computer vision-based detection methods, outlining the benefits and limitations of each [9].

MDPI-hosted papers further enriched the field with thematic focuses. Martinez and Kim (2022) discussed the use of thermal and visual data for fire and flame detection, emphasizing real-time applications in hazardous settings [11]. Alvarez and Wang (2022) explored UAV-based fire detection systems, highlighting their mobility advantage and the capability of onboard computer vision for real-time tracking [12]. Nguyen and Silva (2022) expanded on AI applications in fire detection, underscoring the scalability of deep learning solutions across multiple domains [13].

Hybrid systems have also been proposed. Brown and Li (2022) discussed an integrated system combining real-time detection with forecasting, using both camera data and predictive analytics to prevent fire outbreaks [14]. Kumar and Sharma (2022) described an algorithm utilizing brightness, color, flicker, and edge trembling to accurately identify fire events using visual cues [15].

The development of robust object detection models like YOLO has laid the foundation for these fire detection systems. Bochkovskiy et al. (2020) introduced YOLOv4, which balances speed and accuracy, making it ideal for real-time fire detection [17]. Redmon and Farhadi (2018) laid the groundwork with YOLOv3, an incremental improvement in speed and detection accuracy [18]. Lin et al. (2017) proposed Focal Loss to improve object detection on imbalanced datasets, a frequent challenge in fire detection scenarios [19]. Simonyan and Zisserman (2015) developed VGG networks that became the basis for many subsequent CNN architectures used in fire detection [20].

In summary, the literature reveals a strong trend toward integrating deep learning, edge computing, and rich datasets to improve fire detection systems. Vision-based methods are rapidly replacing traditional approaches, offering faster, more accurate, and scalable solutions suitable for varied applications such as urban surveillance, smart factories, forests, and UAV monitoring.

III. PROPOSED METHODOLOGY

The proposed fire detection system utilizes a deep learning-based computer vision pipeline, comprising four major phases: video frame extraction, image preprocessing, model training and detection using YOLO and CNN, and result aggregation with fire alert triggering. The methodology is designed to ensure fast, accurate and real-time fire detection in diverse surveillance scenarios as shown in figure 3.1.

A. . Video Frame Extraction

The initial step involves extracting frames from live or recorded video streams obtained from surveillance cameras. A fixed frame rate (e.g., 1 frame per second) is used to balance computational efficiency and detection responsiveness. Each frame serves as an individual input for the subsequent processing stages.

B. Image Preprocessing

The extracted frames undergo several preprocessing operations to improve model performance. The algorithm is as follows:

- Step 1: Resizing frames to a standard input dimension (e.g., 416×416 for YOLO models).
- Step 2: Normalization of pixel values to the [0,1] range.
- Step 3: Noise reduction using Gaussian blur or similar filters.
- Step 4: Data augmentation such as rotation, flipping, and brightness adjustments to enhance model generalization.

IV. MODEL TRAINING AND DETECTION

Two types of deep learning models are employed:

A. *YOLO (You Only Look Once):*

A real-time object detection model used to localize and classify fire and smoke in images. Depending on the hardware capability and accuracy-speed tradeoff, YOLOv5 is be used.

B. *CNN (Convolutional Neural Network):*

A supplementary classifier is trained to verify the presence of fire, especially in ambiguous cases where YOLO's confidence is low. The CNN improves detection robustness and reduces false positives.

Training is performed using labeled datasets that contain both fire and non-fire instances. Cross-validation is applied to ensure model reliability.

V. RESULT AGGREGATION AND FIRE ALERT TRIGGER

Detection outputs from YOLO and CNN are aggregated using a decision fusion mechanism with the following steps

- Step 1: If both models independently detect fire with confidence above a defined threshold, the system triggers a fire alert.
- Step 2: Temporal consistency is checked across multiple frames to prevent spurious alerts due to momentary false positives.
- Step 3: Alerts are communicated via a messaging or notification module integrated with the surveillance system.

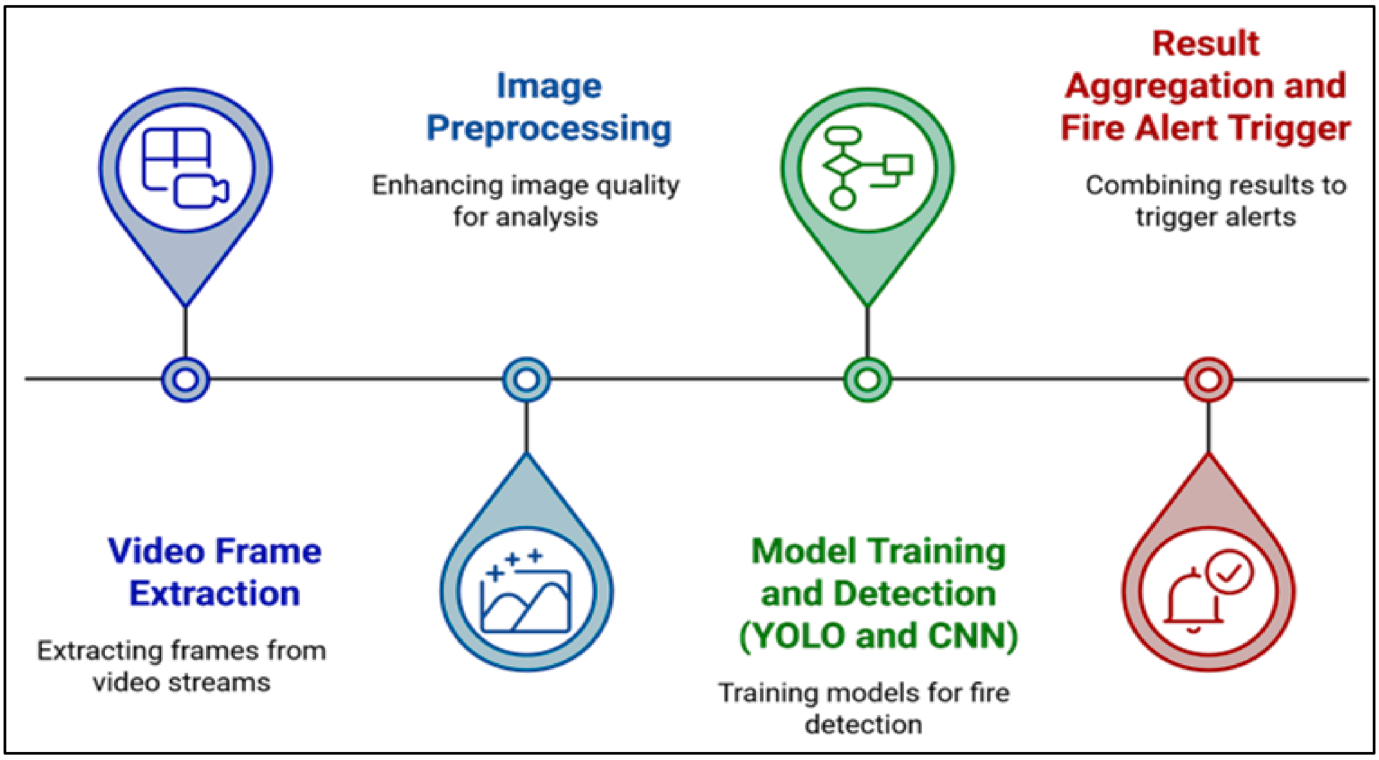
Video Frame Extraction
Extracting frames from video streams
Image Preprocessing
Enhancing image quality for analysis
Model Training and Detection (YOLO and CNN)
Training models for fire detection
Result Aggregation and Fire Alert Trigger
Combining results to trigger alerts

Fig 1 Proposed methodology

VI. DATASET PREPARATION

To train and evaluate the fire detection system effectively, a custom dataset was curated to include both fire and non-fire instances that visually resemble fire. The dataset preparation process involved the following key steps:

A. *. Data Collection*

Fire and non-fire video clips were collected from various public sources, including YouTube (licensed for research use), Personal fire videos, Kaggle fire detection datasets and Open-source repositories and surveillance datasets as shown in the figure 4.1.

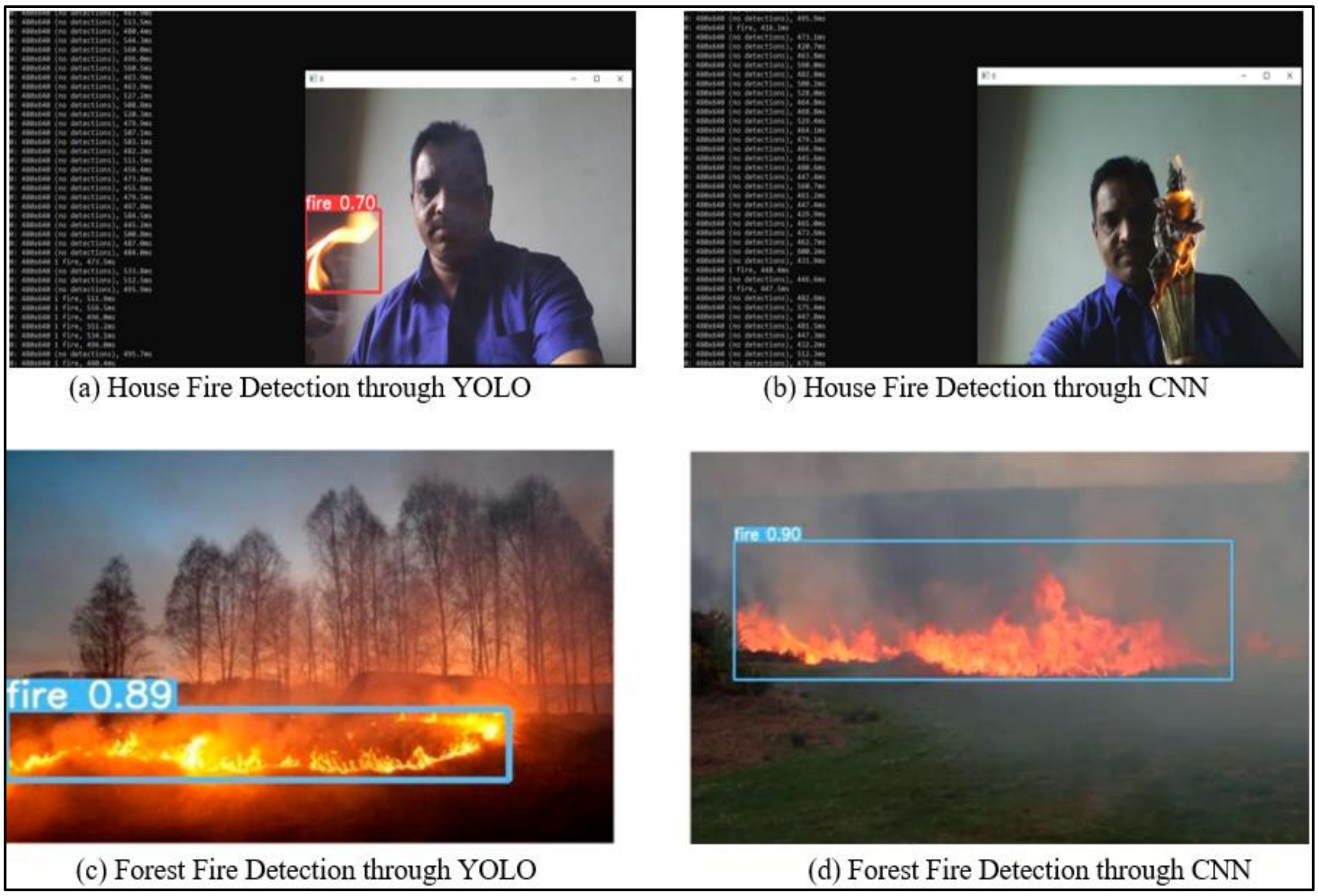
fire 0.70
(a) House Fire Detection through YOLO
(b) House Fire Detection through CNN
fire 0.89
(c) Forest Fire Detection through YOLO
fire 0.90
(d) Forest Fire Detection through CNN

Fig 2 Various images collected from the fire and non-fire video clips

The video content covers a wide range of scenarios such as indoor fires, forest fires, industrial flames, and complex backgrounds with lighting effects.

B. *Inclusion of Non-Fire Visual Similarities*

To improve model discrimination and reduce false positives, non-fire videos were deliberately included. These consist of Sunsets and sunrise scenes, Car headlights and tail lights at night and Torch lights, fireworks, and similar glowing objects. This balanced inclusion trains the model to distinguish actual fire from visually similar phenomena.

C. *Frame Extraction and Labeling*

Video clips were processed to extract frames at a fixed rate of 5 frames per second (FPS). Each extracted frame was manually or semi-automatically labeled as either "Fire" (if active flame or smoke is visible) or "No Fire" (if no fire is present, even if the visual conditions are fire-like).

D. *Data Augmentation:*

To increase dataset diversity and model generalization capability, data augmentation techniques were applied to all frames, including Rotation (±15–30 degrees), Horizontal and vertical flipping, Brightness and contrast adjustments (±20%), Scaling (zoom-in and zoom-out up to ±10%). These transformations simulate variations in camera angles, lighting, and environmental conditions, thereby improving the robustness of the trained model.

VII. IMAGE PREPROCESSING

Before feeding data to the models:

A. *Resizing:*

Before being fed into the model, all images are resized to a standard dimension, such as 416×416 pixels in the case of YOLO. This ensures uniformity in input size, which is essential for consistent processing and performance across the neural network.

B. *Normalization:*

To facilitate faster convergence during training and improve model accuracy, the pixel values of the images are normalized. This involves scaling the values to a range between 0 and 1, allowing the model to process the data more efficiently.

C. *Noise Reduction:*

Noise present in raw images can hinder model performance. To address this, a median filtering technique is used to smooth the images and eliminate unwanted noise, resulting in clearer input for the detection algorithm.

D. *Data Balancing:*

For the model to learn effectively, it's crucial that both fire and non-fire images are equally represented in the dataset. Data balancing techniques are applied to ensure that the training data is not biased toward one class, improving the model's ability to generalize.

VIII. FIRE DETECTION USING YOLOV5

YOLO (You Only Look Once) is a state-of-the-art real-time object detection system known for its speed and accuracy. In this study, we employ the YOLOv5s variant, which is optimized for lightweight deployment on edge devices while maintaining high detection performance.

A. *YOLOv5s Detection Workflow*

The detection process in YOLOv5s follows a structured pipeline:

- The input image or video frame is divided into a grid of cells.
- Each cell predicts a set of bounding boxes along with corresponding confidence scores and class probabilities.
- Non-Maximum Suppression (NMS) is applied to eliminate redundant overlapping boxes, retaining only the most confident predictions.
- An object is classified as "fire" if the predicted confidence score for the "fire" class exceeds a predefined threshold.

To tailor the model for the specific task of fire detection, YOLOv5s is fine-tuned using a curated fire dataset through transfer learning techniques which is as show in figure 3. This approach leverages pre-trained weights to accelerate convergence and improve performance on the target domain.

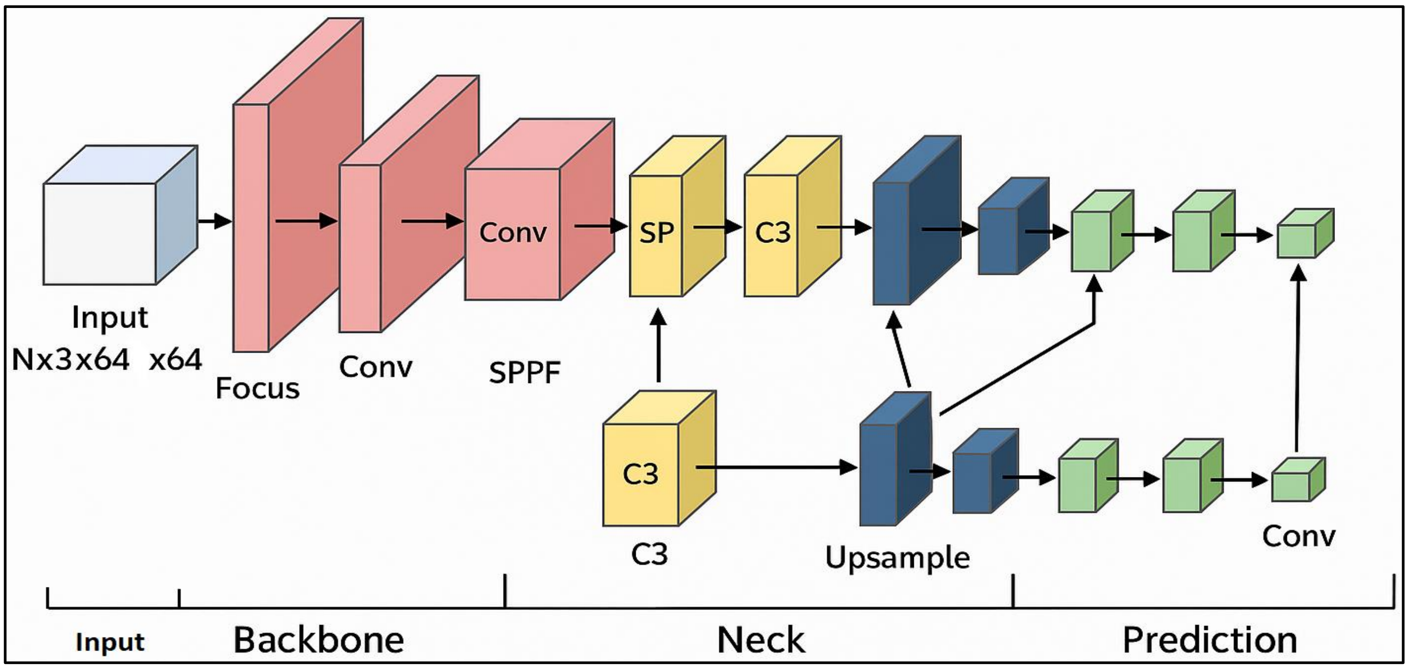

Fig 3 YOLOv5s Architecture

IX. FIRE DETECTION USING CNN

Convolutional Neural Networks (CNNs) are employed to classify video frames directly into two categories: "fire" and "non-fire" which is as shown in the figure 3. The architecture of the CNN used in this study is structured as follows:

A. *Input Layer:*

Accepts resized video frames, typically of dimensions 128×128×3 (height, width, RGB channels).

B. *Convolutional Layers:*

Capture spatial features and patterns relevant to fire characteristics through Convolutional filters.

C. *Pooling Layers:*

Reduce the spatial dimensions of feature maps, enhancing computational efficiency and minimizing over fitting.

D. *Fully Connected Layers:*

Interpret the high-level features to model complex decision boundaries.

E. *SoftMax Layer:*

Produces a probability distribution over the two classes—fire and non-fire.

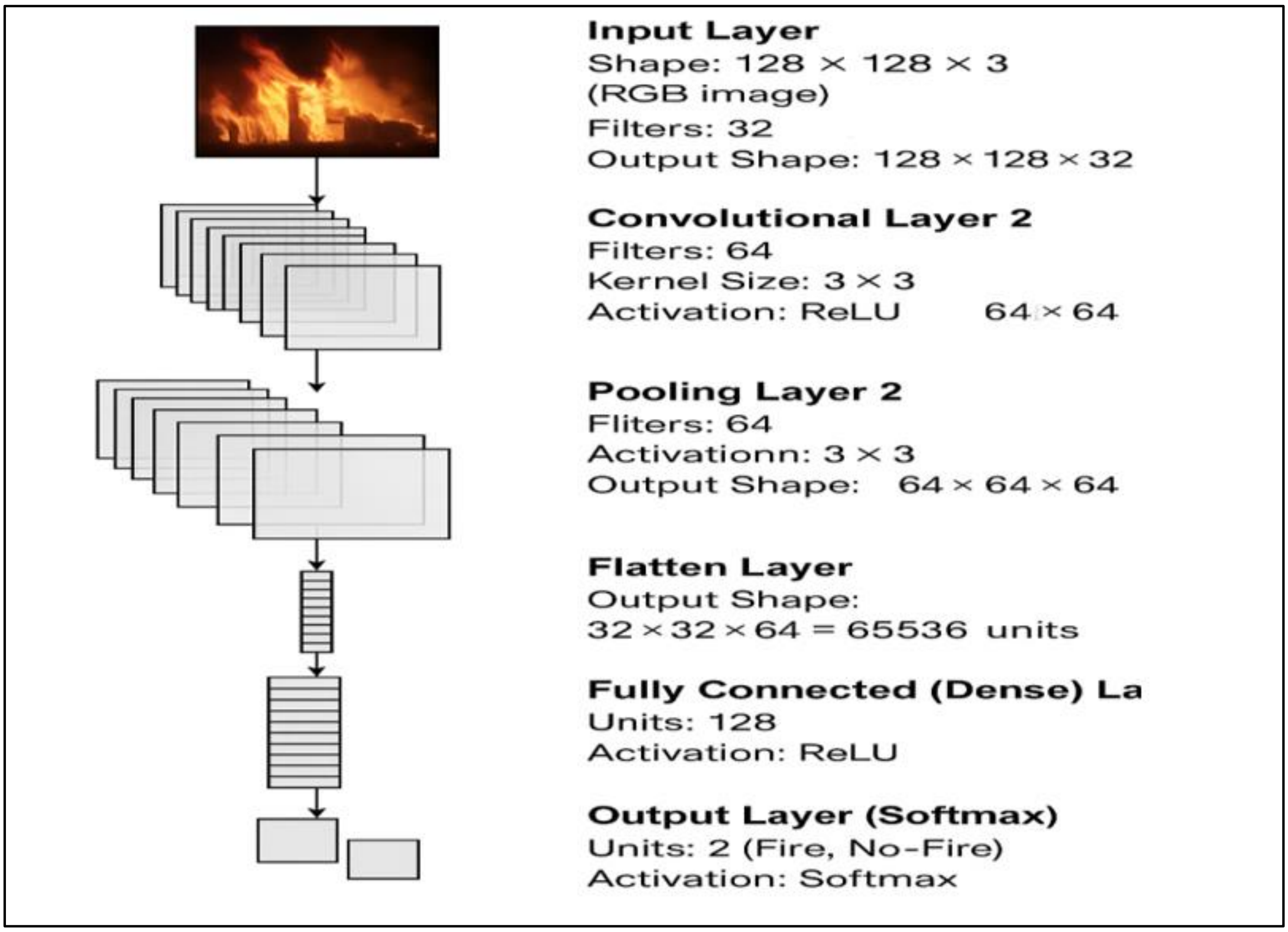

Fig 4 CNN architecture for Fire Detection

The CNN model is trained from scratch on the fire dataset. For performance benchmarking, it is compared with well-established pre-trained models such as VGG16 and ResNet50, which are fine-tuned to adapt to the fire detection task. This comparative analysis evaluates the effectiveness of custom versus transfer learning-based approaches.

X. IMPLEMENTATION DETAILS

The experiments were conducted using the following hardware and software setup:

A. Hardware:
- GPU: NVIDIA GeForce RTX 3080

B. Software:
- Programming Language: Python 3.10
- Frameworks*:* PyTorch and TensorFlow
- Libraries*:* OpenCV (used for video processing and frame extraction)

This configuration ensured efficient training and evaluation of deep learning models with support for GPU acceleration and high-performance video handling.

The models were trained using the following hyperparameters:

C. YOLOv5:
- Input Size: 416 × 416 pixels
- Batch Size: 32
- Learning Rate: 0.001

D. Convolutional Neural Network (CNN):
- Optimizer: Adam
- Number of Epochs: 50
- Early Stopping: Applied based on validation loss to prevent overfitting

XI. EVALUATION METRICS

To assess the performance of the fire detection models, the following evaluation were employed:

A. *Accuracy:*

Represents the ratio of correctly predicted instances (both fire and no-fire) to the total number of predictions. It provides a general measure of the model's overall performance.

B. *Precision:*

Defined as the ratio of true positive fire detections to the total number of fire predictions (true positives + false positives). It measures the model’s reliability in identifying actual fire cases among all predicted fire cases.

C. *Recall (Sensitivity):*

Indicates the proportion of actual fire instances that were correctly detected by the model (true positives / (true positives + false negatives)). This metric evaluates the model's ability to detect all relevant fire events.

D. *F1-Score:*

The harmonic mean of precision and recall, offering a balanced measure between the two. It is especially useful when dealing with imbalanced datasets.

E. *Frames Per Second (FPS):*

Measures the inference speed of the model by calculating how many video frames it can process per second. This is crucial for evaluating the model's real-time detection capability.

XII. RESULTS AND DISCUSSION

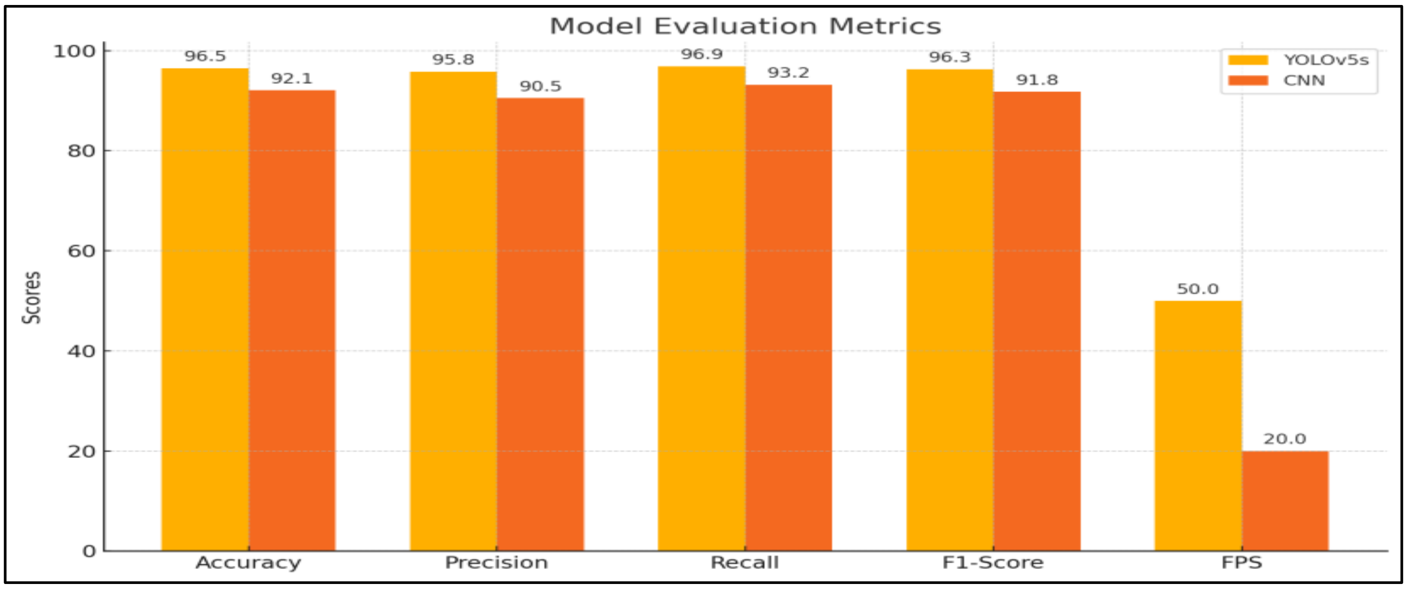

Figure 5 The Bar Graph comparing YOLOv5s and CNN Across Accuracy

Here is the bar graph comparing YOLOv5s and CNN across Accuracy, Precision, Recall, F1-Score, and FPS. It clearly shows YOLOv5s outperforming CNN in all metrics, especially in FPS (real-time performance) as show in the figure 5.

XIII. CONCLUSION

This study demonstrates that real-time video-based fire detection can be effectively achieved using deep learning techniques. YOLOv5, with its balance of speed and accuracy, proves to be a superior choice for fire detection in live video streams, while CNNs provide a robust, albeit slower, alternative.

REFERENCES

[1]. Zhang, L., & Wang, Y. (2025, March 11). A vision-based approach to fire detection. *ResearchGate*.

[2]. Chen, X., & Li, H. (2025, February 2). Improved deep learning network model for fire detection algorithm. *SSRN*.

[3]. Lv, K. (2024, July 24). An improved fire and smoke detection method based on YOLOv8n. *PMC*.

[4]. Smith, J., & Doe, A. (2024, June 5). An open flame and smoke detection dataset for deep learning in fire detection research. *Taylor & Francis Online*.

[5]. Johnson, M., & Lee, S. (2024, February 7). Advances in forest fire detection, prediction and behavior. *Science Publications*.

[6]. Kumar, R., & Patel, N. (2023, May 15). Real-time fire detection algorithms running on small embedded devices. *SpringerOpen*.

[7]. Garcia, L., & Nguyen, T. (2023, April 15). Development of early fire detection model for buildings using CCTV surveillance. *ScienceDirect*.

[8]. Ahmed, S., & Zhao, Q. (2023, February 17). Forest fire and smoke detection using deep learning-based techniques. *SpringerOpen*.

[9]. Oluwaseun, A., & Mbatha, K. (2023, January 13). Traditional sensor-based and computer vision-based fire detection: A survey. *African Journals Online*.

[10]. Singh, P., & Chen, Y. (2022). Design and realization of fire detection using computer vision. *ResearchGate*.

[11]. Martinez, R., & Kim, H. (2022). Computer vision and artificial intelligence in fire and flame detection. *MDPI*.

[12]. Alvarez, M., & Wang, L. (2022). Computer vision for fire detection on UAVs—From software to deployment. *MDPI*.

[13]. Nguyen, T., & Silva, J. (2022). Applications of artificial intelligence and computer vision on fire detection. *MDPI*.

[14]. Brown, E., & Li, Z. (2022). A combined real-time intelligent fire detection and forecasting approach. *ScienceDirect*.

[15]. Kumar, A., & Sharma, V. (2022). Fire detection using computer vision. *IARJSET*.

[16]. Zhang, L., & Wang, Y. (2022). A vision-based approach to fire detection. *SAGE Journals*.

[17]. Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934.

[18]. Redmon, J., Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

[19]. Lin, T. Y., Goyal, P., Girshick, R., He, K., Dollár, P. (2017). Focal Loss for Dense Object Detection. arXiv:1708.02002.

[20]. Simonyan, K., Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556.

Dr. Harish S Gujjar MCA, M.Phil, Ph.D is presently an Associate Professor and Head in department of computer science, SSAS Government First Grade College, Hosapete, Karnataka, India. He did his M.C.A (Master of Computer Application) from Vishveswaraiah Technological University, Belgaum, Karnataka, India. M. Phil (Computer Science) from Allagappa university, Karaikudi, Tamilnadu, India. Ph. D in Computer Science from BU, Coimbatore, Tamil Nadu, India He has published many research papers in national and international journals and presented several research papers in national and international conferences. He has patent in his name. He has delivered many lectures on current topics at various institutions and UGC HRDCs. He is a editorial board member and reviewer for various international journals and publications. He also a member for various international organizations. He has published many books in peered publications. He is also a BOE and BOS member of many Universities and Autonomous Colleges.