Optimized Real-Time Video-Based Wildfire Detection on Embedded Devices Using Enhanced YOLOv5

ASSOUMANA SOULEY HADIZA DITE MAA, MKOUKA BTISSAM, and ZAHEDI CHAYMAA

Abstract

Early wildfire detection is crucial to minimize property damage and protect human lives. However, traditional systems often rely on expensive equipment or suffer from delayed response times. This project proposes a low-cost and efficient real-time fire detection solution based on the YOLOv5 deep learning model, deployed on a Raspberry Pi. The primary objective is to design a lightweight, autonomous system capable of detecting fire outbreaks in various environments without the need for complex infrastructure.

The methodology involves training a YOLOv5 model on a dataset containing images with and without fire, enhanced through data augmentation techniques to improve robustness. Once optimized, the model is deployed on a Raspberry Pi, taking into account its limited computational resources and to accommodate this, the model is compressed and converted to INT8 ONNX format, enabling faster inference and reduced memory footprint. The system's performance was evaluated through tests conducted in diverse scenarios.

In this paper, we introduce a new fire detection methodology based on YOLOv5 designed to address these challenges. This system combines high detection accuracy with real-time performance, representing a significant advancement in fire detection technology. Testing the system showed impressive results—achieving a detection accuracy of 87% , along with high precision and controlled false positive rates. Furthermore, deploying the solution on a Raspberry Pi 4 ensures cost-effectiveness and energy efficiency, enabling broader and more accessible deployment.

This system is not only vital for minimizing fire-related property damage but also plays a critical role in saving lives by providing timely alerts during fire emergencies.

1.
E-mail:

1. Introduction

In recent years, the devastating impact of wildfires on both human life and property has increasingly emphasized the need for rapid and reliable fire detection systems. Fires can break out in residential areas, industrial zones, and natural landscapes, often without warning, making early detection essential to limit their consequences. In response, there is growing demand for intelligent systems capable of recognizing fires in real time and triggering timely alerts to support faster evacuations and intervention.
At the intersection of artificial intelligence (AI) and embedded systems, a new wave of innovation is emerging. Intelligent embedded devices such as the Raspberry Pi are becoming powerful enough to host lightweight AI models, enabling real-time decision-making directly on the edge. This advancement makes it possible to develop autonomous and low-cost fire detection systems that do not depend on traditional physical sensors (e.g., heat, smoke) or remote servers. Instead, they rely on computer vision and deep learning, offering greater flexibility and improved accuracy in diverse environments.
However, challenges remain. Deep learning models such as Convolutional Neural Networks (CNNs) have demonstrated high accuracy in image recognition tasks but often require considerable computational resources. Their deployment on resource-constrained platforms like Raspberry Pi demands careful model optimization to ensure responsiveness and energy efficiency. Moreover, many existing fire detection methods lack the real-time capability necessary to prevent the spread of fire effectively.
To address these challenges, we explore whether it is possible to design a real-time, reliable, and low-cost fire detection system based on computer vision, running entirely on an embedded device. In this paper, we introduce an optimized methodology using YOLOv5, a state-of-the-art object detection model known for its speed and accuracy. The proposed system is designed to identify flames from live video feeds and issue immediate alerts while maintaining high performance on low-power hardware.

Our contributions are as follows:

1. We build and preprocess a custom dataset specifically tailored for visual fire detection.

2. We train and optimize YOLOv5 model for efficient deployment on Raspberry Pi 4.

3. We implement and evaluate a complete working prototype in diverse real-world environments, achieving 87% detection accuracy and maintaining low false positive rates.

4. We enhance the prototype with embedded alert mechanisms, including a buzzer, a red LED for visual indication, and automated SMS notifications via the Infobip API to ensure timely alerts in case of fire detection.

The following is how the remaining work is structured. Section 2 presents some of the most recent research in the field of AI in fire detection. The proposed framework is presented in Sect. 3. In Sect. 4, experimental evaluation is offered. Section 5 brings this effort to a close.

2. Art of state

Fire detection through computer vision has seen significant advancements in recent years, driven by the emergence of deep learning models such as the YOLO (You Only Look Once) family. In particular, YOLOv5 has become a popular choice due to its balance between accuracy and speed. However, deploying such models on resource-constrained embedded systems still presents multiple challenges, which has led to extensive optimization efforts between 2020 and 2025.
Ma et al. (2024) [2] proposed an enhanced version of YOLOv5s by integrating a lightweight Transformer module into the backbone and adopting a new loss function (-CIoU) to improve localization accuracy. Their semi-automatic data annotation strategy significantly reduced dataset preparation costs, achieving a precision of 83.9%.
Xu Li (2022) [Xu et al.] introduced "Light-YOLOv5", a streamlined variant that replaces standard blocks with SepViT (separable transformers) modules and incorporates a global attention mechanism. Their model reached speeds exceeding 91 FPS while significantly lowering computational complexity.
Islam Habib (2023) [Islam and Habib] focused on detecting small-scale fires by integrating spatio-temporal (3D) convolutions and a tailored feature pyramid network. Although slower (8 FPS), their model achieved high performance with a mean Average Precision (mAP) of 90.5%.
Yar et al. (2023) [Yar et al.] optimized the architecture for smart environments by introducing a "stem" module to down sample the input early and adapting the FPN pyramid. These design choices allowed their model to reach an impressive mAP of 92.2% at 45 FPS on GPU.
DFrom an embedded systems perspective, Zhang et al. (2022) [He et al.] designed a custom solution targeting ARM Cortex-A53 architectures. By applying pruning and INT8 quantization, their system maintained an 88% mAP while running at 30 FPS, demonstrating the feasibility of effective fire detection on low-power hardware.
Before the deep learning era, Pritam Dewan (2002) [1] developed a classical method combining YCbCr color space filtering with spatiotemporal image analysis. Although dated, their approach remains relevant as a preprocessing step to filter out false positives.
In general, these studies highlight the growing interest in optimizing YOLOv5 for fire detection, with a particular focus on model simplification, integration of attention mechanisms, and adaptation to the constraints of embedded platforms. This research landscape provides strong motivation and guidance for the development of our own embedded fire detection system, centered on a careful trade-off between lightweight design, detection accuracy, and real-time responsiveness.

3. Methodology

3.1 Proposed Approach

This project aims to develop an automatic fire detection system based on computer vision, using YOLOv5's capabilities to perform fast and accurate detection. The objective is to prepare an optimized solution that can eventually be deployed in low-power embedded systems. To this end, we used YOLOv5n, the lightest version of the YOLOv5 family, well-suited for environments with limited computational resources.

Our approach includes the following steps: selecting an annotated dataset, training the model, exporting it to the int8ONNX format, and then applying post-training quantization in order to reduce the model size and evaluate its behavior in an environment simulating an embedded platform. The final system also integrates additional features for real-world deployment, including a buzzer for audio alerts, a red LED for visual warnings, and an SMS notification module using the Infobip API to ensure rapid remote alerts when fire is detected.

3.2 System Architecture

The system is based on a classic real-time detection architecture, now enhanced with an embedded alert mechanism, consisting of:

- An image or video loading module, which enables the model to process a variety of visual sequences.
- An optimized YOLOv5n-based processing module, responsible for frame-by-frame analysis to detect fire.
- A result display module, used to visualize the predictions with overlays (bounding boxes and labels).
- An embedded alarm module that triggers a buzzer or speaker when fire is detected, providing an immediate audible alert.
- A red LED indicator that lights up in the presence of fire, offering a visual warning signal.
- A remote notification system that sends an SMS alert via the Infobip API, allowing for rapid response even when users are not physically present.

This alarm functionality is controlled through GPIO pins when deployed on platforms such as the Raspberry Pi. The system was tested in a desktop environment with simulated alarm activation, with the perspective of full integration on embedded hardware for real-world deployment.

3.3 Implementation Details

The model training was conducted using an open-source annotated dataset found in Roboflow, containing fire images captured in various environments (indoors, forests, urban areas, etc.). Training was performed using YOLOv5n with the Ultralytics library on a GPU-equipped environment (approximately 100 epochs, batch size of 16).
The final model was exported in int8ONNX format to ensure compatibility with a wide range of deployment environments. Post-training quantization was applied to reduce the model size by converting the weights to INT8 format, enabling simulation of execution on low-power devices.

3.4 Experimental Setup

The tests were conducted on a personal computer (i5 CPU, 16 GB RAM), without using an embedded platform. The quantized model was evaluated using ONNX Runtime, allowing the measurement of its performance in terms of detection accuracy (mAP, precision, recall) as well as execution speed (number of images processed per second).
These tests serve as a proof of concept prior to potential deployment on a platform such as the RaspberryPi.

4. Results and Analysis

4.1 Performance metrics : Recall and Precision

To evaluate the performance of the fire detection system, we focus on two essential metrics: **Recall** and **Precision**. These indicators provide valuable insights into the model's effectiveness in identifying actual fire events and minimizing false detections.

Recall reflects the system's ability to correctly detect all real fire instances. It is defined as the proportion of true positive detections to the total number of actual fires. A high recall value indicates that the model rarely misses true fire cases, which is critical in emergency scenarios. Recall is calculated as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{1}$$

Where:

- TP (True Positives): the number of correctly detected fire instances.
- FN (False Negatives): the number of actual fire events that the model failed to detect.

Precision, on the other hand, measures the accuracy of the fire alerts produced by the system. It is the ratio of true positive detections to the total number of instances identified as fire. A high precision score indicates that most alerts are relevant, thus minimizing false alarms. Precision is computed using the formula:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

Where:

- FP (False Positives): the number of non-fire events that were incorrectly identified as fire.

These metrics together offer a comprehensive view of the model's performance in real-world deployment scenarios.

4.2 Results

The evaluation of our approach, based on the YOLOv5n model trained on an annotated dataset from the web for fire detection, revealed overall promising performance. The model achieved an mAP@0.5 of 91%, with a precision of 87.8% and a recall of 84.2%, indicating a strong ability to accurately identify fire outbreaks while maintaining a low false positive rate. Figure 1 below illustrates the training metrics, including precision and recall, which further demonstrate the model's effectiveness. Post-training quantization, following conversion to the int8ONNX format, reduced the model size from 27 MB to 7.2 MB, a crucial step toward future deployment on embedded systems like the Raspberry Pi. This compression did not result in significant performance loss, highlighting the robustness of YOLOv5n under light optimizations. The training metrics, along with the specific precision and recall charts shown in Figure 1, support these findings.

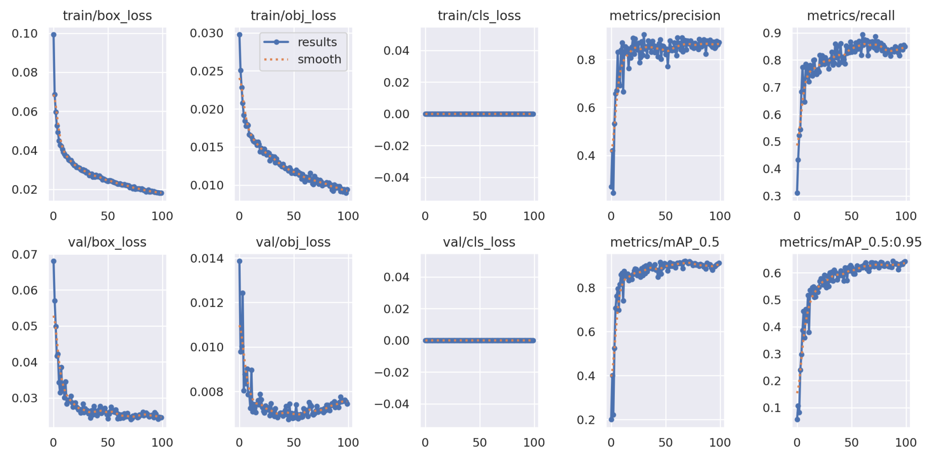

Figure 1: Training Metrics, Precision, and Recall for our YOLOv5n Model.

To further validate the performance of our system, we present in Figure 2 several visual examples from both the training and testing phases. These examples illustrate the model's ability to accurately detect fire under different conditions, including indoor scenes, forest fires, and urban environments. The bounding boxes and labels clearly show correct identification of fire regions. These qualitative results reinforce the quantitative metrics previously discussed, and confirm the effectiveness of the YOLOv5n-based solution even after model compression and deployment optimization.

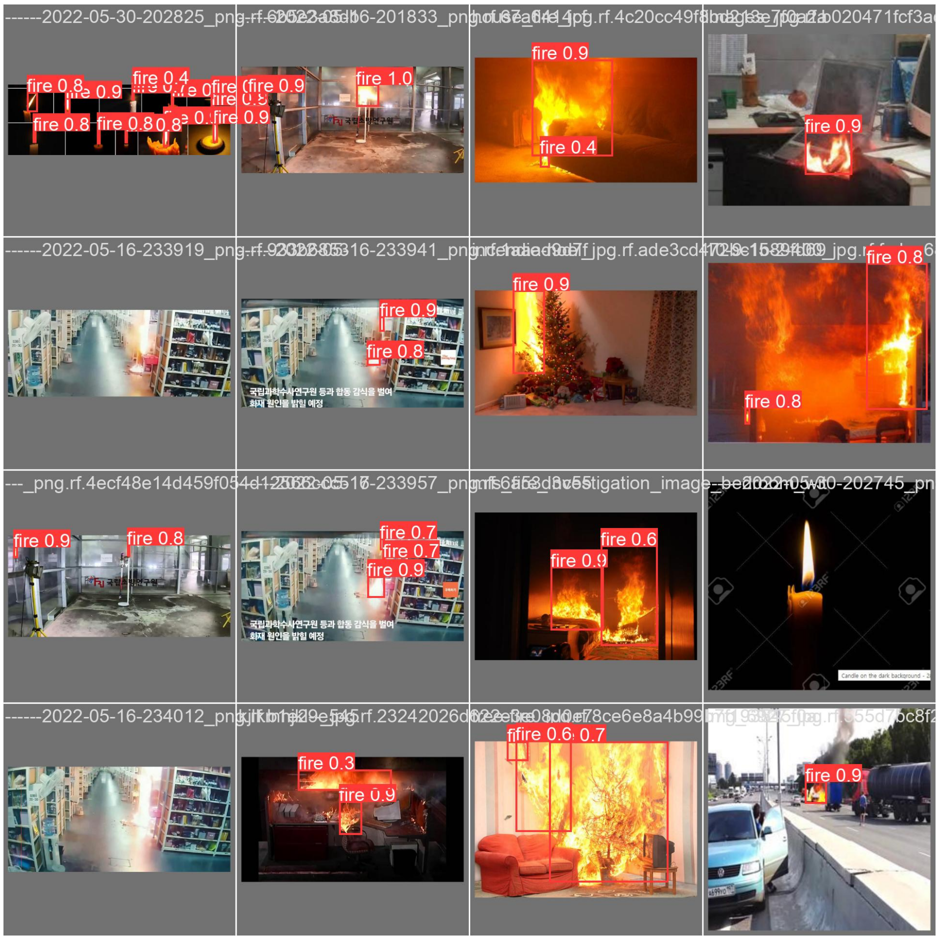

Figure 2: Qualitative results showing fire detection on selected training and testing images using the YOLOv5n model.

To further assess the model's practical applicability, we conducted real-time fire detection tests using a webcam input. As illustrated in Figure 3, the YOLOv5n model was able to successfully detect fire outbreaks with minimal latency, demonstrating its potential for deployment in real-world surveillance systems. The model's lightweight structure and optimized int8ONNX format allowed smooth execution even on hardware with limited computational capacity.

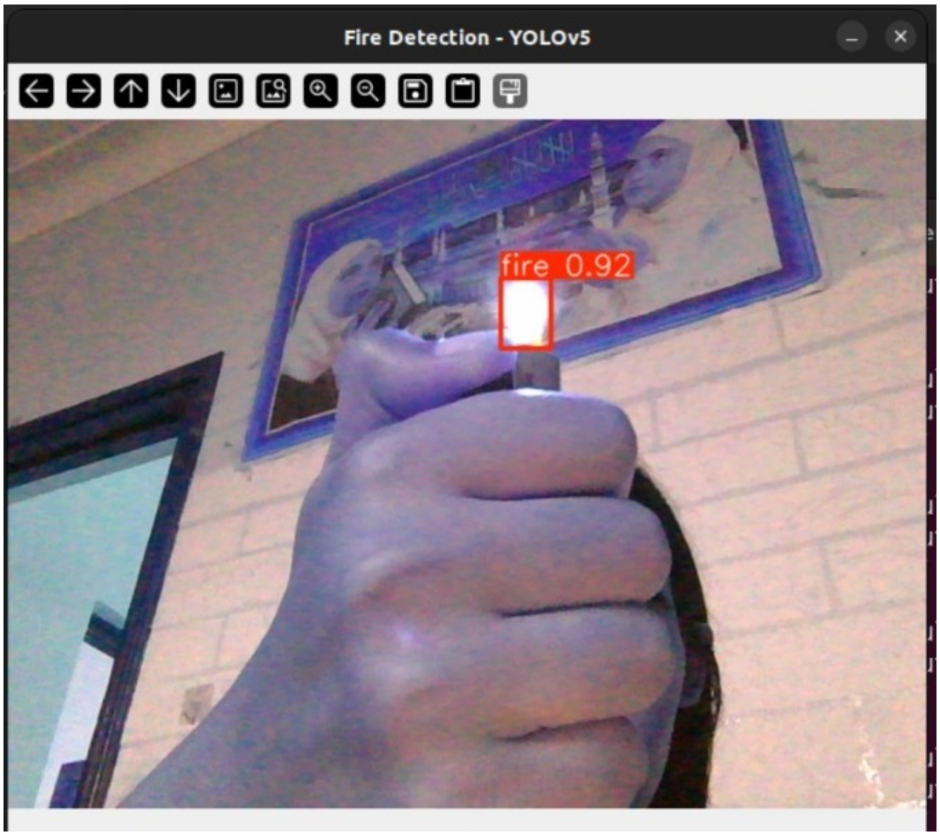

Figure 3: Qualitative results showing Real Time fire detection.

5. Discussion

The evaluation of our approach, based on the YOLOv5n model trained on an annotated dataset sourced from the web, revealed overall satisfactory fire detection performance. The model achieved an mAP@0.5 of 91%, with a precision of 87.8% and a recall of 84.2%, indicating a strong ability to accurately identify fire outbreaks while maintaining a low false positive rate. Post-training quantization of the model, after conversion to the int8ONNX format, reduced its size from 27 MB to 7.2 MB, which is a key step towards a deployment on embedded systems with the Raspberry Pi. This compression did not result in significant performance loss, demonstrating the robustness of YOLOv5n under light optimizations.
Compared to state-of-the-art results, our model's performance lies between that of Ma et al. (mAP 74%) and Yar et al. (mAP 92.2%), offering an interesting trade-off between model lightweightness and accuracy. Unlike some works integrating complex modules (such as SepViT or lightweight Transformers), our solution favors a simple yet optimized architecture, achieving good results without requiring high hardware resources.
However, some limitations remain. On the one hand, the dataset used, while annotated, comes from diverse sources without strict normalization, which may affect the model's

generalization.
In summary, our approach provides a solid and lightweight foundation for embedded fire detection, with performance comparable to heavier models in the literature, while still leaving room for improvement in certain technical and experimental aspects.

5.1 Conclusion

In this work, we explored the use of the YOLOv5n model for fire detection from images, with the goal of future deployment on embedded systems such as the Raspberry Pi. After training the model on an annotated dataset sourced online, we applied several optimizations, including conversion to the ONNX format and post-training quantization. These steps significantly reduced the model size—from 27 Mo to 7.2 Mo—while maintaining high performance, achieving an mAP@0.5 of 91%, precision of 87.8%, and recall of 84.2%.

These results highlight the relevance of lightweight models like YOLOv5n for resource-constrained embedded applications. Our contribution stands out due to its simple implementation, competitive performance, and focus on practical fire detection scenarios, making it accessible for a wide range of devices.

For future work, several avenues can be explored:

- Integrating attention modules or lightweight transformers to further improve precision without sacrificing model compactness.
- Creating or enriching a more homogeneous dataset with standardized annotations covering a wider variety of fire situations (indoor, outdoor, urban, forest fires, etc.).

Ultimately, this study demonstrates that it is possible to combine efficiency, compactness, and simplicity in designing fire detection systems, thereby contributing to enhanced safety across diverse environments through embedded artificial intelligence.

References

[1] Computer vision based fire detection in color images | request PDF.

[2] https://ieeexplore.ieee.org/document/10973447.

[He et al.] He, H., Zhang, Z., Jia, Q., Huang, L., Cheng, Y., and Chen, B. Wildfire detection for transmission line based on improved lightweight YOLO. 9:512–520.

[Islam and Habib] Islam, A. and Habib, M. I. Fire detection from image and video using YOLOv5.

[Xu et al.] Xu, H., Li, B., and Zhong, F. Light-YOLOv5: A lightweight algorithm for improved YOLOv5 in complex fire scenarios.

[Yar et al.] Yar, H., Khan, Z. A., Ullah, F. U. M., Ullah, W., and Baik, S. W. A modified YOLOv5 architecture for efficient fire detection in smart cities. 231:120465.