Received January 20, 2025, April 15, 2025, date of publication May 1, 2025.

Digital Object Identifier 10.21608/ijaici.2025.337982.1003

Fire Detection Model Based on the Yolov8 Model for the Industrial Field

Mustafa Abdul Salam [1], Alsayed Badr [2], Aya Abdelhamed Elbohoty [2], Eman Mohamed [3]

[1] Department of Artificial Intelligence, Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt.

[2] Department of Scientific Computing, Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt

[3] Benha University, Benha, Egypt.

Corresponding author: (e-mail mustafa.abdo@fci.bu.edu.eg)

Abstract

ABSTRACT This paper proposed a model to detect fires in the industrial field using You Only Look Once Version 8 (YOLOv8) framework. The proposed model is based on three primary stages which are data pre-processing, feature selection, and evaluating the results using a variety of metrics. Images are resized, enhanced, noise is reduced, and videos and images are labeled with bounding boxes surrounding the fires during the data pre-processing step. The YOLOv8 model's speed and accuracy make it the preferred choice for feature selection. The performance of the proposed model is assessed using a variety of metrics, including accuracy, precision, and recall. Furthermore, the suggested model is trained in a real-time system that is capable of processing camera feeds in real time. When a fire is detected, the building's fire alarm should go on to alert people and tell them to evacuate. The experiment's findings show that the recommended model produced results with 98.1% accuracy, 98.9% precision, 95.3% recall, and 98.1% mAP. Finally, the proposed model is contrasted with existing methods on the same dataset.

INDEX TERMS: YOLOv8, Deployment, deep learning, fire detection, Integration

1. Introduction

Fires have the potential to cause serious financial harm in addition to endangering lives. In 2020, there were 1153 forest fires in China, seven of which were big ones, according to incomplete statistics. It damaged people's lives and property and resulted in an economic loss of CNY 162.19 million. As a result, fire detection is crucial to safeguarding forest resources as well as the lives and property of humans. At the moment, there are three fundamental ways for detecting fires: image processing technology, sensor detection, and manual inspection. Conventional hand inspection and sensor detection are sensitive to external factors such as temperature, humidity, wind speed, and spatial location, and they respond slowly and tire easily. As a result of this, false alarms frequently occur, and early fire detection is challenging. The precision of image processing algorithms has been continually improving while maintaining their advantages of low cost and great efficiency [1]. As a result, a number of scientists have used image processing methods to identify flames. Conventional image processing methods for fire detection usually use hand-picked features, like color [2], texture [3], and geometric features [4], to divide up flames. These fire segments are then used to classify and match the images using machine learning algorithms. However, using manually designed feature extraction, standard image processing algorithms cannot fulfill the requirements for model generalization capability and robustness in real engineering due to the complexity of the fire environment. The redundancy and interference brought on by the manual extraction of image features can be efficiently overcome by deep learning target detection, which can automatically extract picture details and images features [5].

Fire detection systems that depend on deep learning approaches rather than feature descriptions are attracting growing attention [6, 7]. Since deep learning-based fire recognition is an improvement over classic image processing-based method. Despite this, most deep learning fire video detection systems available today require computers equipped with robust CPUs and GPUs for accelerated processing, in addition to a few embedded platforms for cloud image recognition. This is a result of deep learning requiring a significant amount of computing power and target recognition emphasizing real-time recognition. They first impact the hull's layout, need a significant amount of wiring, and take up a lot of space and money. It's necessary to have good network coverage [8].

The aforementioned research leads us to conclude that the present fire detection algorithms have two main limitations: an excessive number of parameters that make the algorithm difficult to compute and a weak resistance to environmental changes that expose the algorithm to false alarms. Due to these factors, an early real-time fire detection system based on the You Only Look Once version 8 algorithm (YOLO v8) for industrial applications is presented in this paper. The usefulness of the model for fire detection in real-world circumstances is demonstrated by the experimental findings that have been supplied and documented. The proposed method shows the benefits of our suggested system and contrasts it with other fire detection systems currently in use. The main contribution of this paper is:

- The model based on You Only Look Once Version 8 (YOLOv8) framework for fire detection in the industrial field is proposed.
- An extensive dataset with fire, smoke, and typical scenes is used, in contrast to other approaches that use a limited number of datasets. Real-world photos and videos gathered from multiple sources are included in the dataset. There are many different types of fire scenarios in the dataset, such as small and big fires, indoor and outdoor fires, and low- and high-light situations.
- Through the application of deep learning to identify fire-specific characteristics, the suggested model may reduce false alarms and prevent needless emergency responses.
- By utilizing the advantages of deep learning algorithms like YOLOv8, the suggested model may increase the accuracy of fire detection when compared to traditional methods. It achieved 98.1% accuracy, 98.9% precision, 95.3% recall, and 98.1% mAP.

The remaining parts of the paper are structured as follows: **Section 2** reviews related literature, while **Sections 3 and 4** outline the proposed algorithm, feature selection, and YOLO model. **Section 5** presents the experimental results, and the final section summarizes the conclusions.

2. Related work

Traditional fire detection methods often rely on flame and smoke sensors, but these approaches are unsuitable for all environments. The growing use of video cameras in public safety systems has enabled the development of machine learning-based fire detection techniques that rely on image analysis. Early vision-based methods primarily relied on handcrafted features such as color, texture, and shape to identify fires. For example, Chen et al. [9] proposed a rule-based approach using RGB and HSI color models to analyze fire behavior across frames. Celik and Demirel [10] introduced a flame pixel classification technique based on the YCbCr color model, while Wang et al. [11] leveraged flame color dispersion to distinguish fire regions. However, these color-based methods are highly sensitive to lighting conditions, shadows, and reflections, leading to frequent false alarms.

To enhance robustness, researchers incorporated additional features such as motion and texture. Borges and Izquierdo [12] developed a probabilistic model integrating color and motion characteristics, while Mueller et al. [13] utilized optical flow to differentiate fire from other moving objects. Foggia et al. [14] proposed a multi-expert system combining color, shape, and motion analysis. Despite these improvements, conventional approaches remain limited by their reliance on hand-crafted features, making them less effective in complex environments.

Recent advancements in deep learning have significantly improved fire detection accuracy. Deep learning models such as convolutional neural networks (CNNs) have been employed to automate feature extraction, reducing false positives and enhancing generalization. For instance, Wang et al. [15] proposed a flame detection system based on KNN background subtraction, while Ghosh et al. [16] combined CNN and recurrent neural networks (RNNs) to improve accuracy in forest fire detection.

Single-stage object detectors, such as the YOLO family of models, have gained popularity due to their high speed and accuracy. Redmon et al. [17] introduced the original YOLO model, which treats object detection as a regression problem. Subsequent versions have refined its architecture, improving precision and efficiency. The YOLO family of algorithms has gained popularity due to its high accuracy and efficiency. YOLO models can be trained using a single GPU, making them accessible to a wide range of developers. The latest iteration, YOLOv8, introduced by Ultralytics [18], builds upon the success of YOLOv5 with architectural improvements for enhanced performance in object detection, image classification, and segmentation.

YOLOv8 [19] adopts an **anchor-free** detection approach, estimating object centers directly instead of relying on anchor boxes. This reduces the number of predictions, accelerating Non-Maximum Suppression (NMS), a crucial post-processing step. YOLOv8 consists of five model variants—YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x—where YOLOv8n is the fastest and smallest, while YOLOv8x is the most precise but computationally intensive.

Key architectural enhancements in YOLOv8 include:

- **C2f Module:** Replaces the previous C3 module for improved feature extraction as shown in FIGURE 1.
- **Backbone Optimization:** The initial 6×6 convolution was replaced with a 3×3 convolution for efficiency.
- **Decoupled Head:** The objectness step is removed for streamlined prediction.

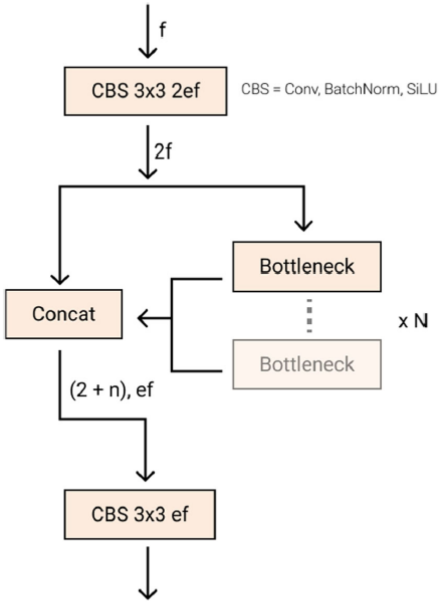

FIGURE 1. Yolov8 C2f Module [19].

These enhancements contribute to YOLOv8's superior performance, making it a robust solution for real-time fire detection applications. This study builds on prior research by integrating YOLOv8 for fire detection in industrial environments, optimizing its performance through fine-tuning and dataset augmentation. With enhanced object localization and reduced false positives, YOLOv8 offers a highly effective approach to fire monitoring and safety.

The choice of an appropriate algorithm depends on the specific requirements of the fire detection system, ensuring it can handle diverse fire scenarios and perform reliably on the dataset. Due to its high accuracy and speed, YOLOv8 is a widely preferred option; however, alternative algorithms may be utilized depending on the system's needs. The YOLOv8 model is trained using deep learning frameworks like TensorFlow or PyTorch, which provide essential tools and libraries for constructing and optimizing neural networks.

Structurally, YOLO consists of 24 convolutional layers followed by two fully connected (FC) layers. Some convolution layers utilize 1×1 reduction layers to decrease the depth of the feature maps efficiently. The final convolution layer outputs a tensor of size (7,7,1024), which is then flattened and passed through two FC layers to generate $7 \times 7 \times 30$ parameters. These parameters are reshaped into (7,7,30), and YOLO utilizes a sum-squared error function to compute the difference between ground truth values and predictions.

The **YOLO** loss function comprises **classification loss, localization loss, and confidence loss**, as described by **Redmon et al. [17]**. These equations form the foundation of **YOLO's object detection framework**, ensuring a balance between classification and localization accuracy. This methodology supports the **YOLOv8-based fire detection system** implemented in this study.

$$\text{Classification loss} = \sum_{j=0}^{m^2} 1_j^{obj} \sum_{i \in class} (p_j(i) - \hat{p}_j(i))^2 \quad (1)$$

$$Localization\ loss = \partial_{record} \sum_{j=0}^{m^2} 0' \quad (2)$$

Where $1_j^{obj} = 1$ if object in j cell, otherwise 0 and $\hat{p}_j(i)$ represents the condition class for i class in j cell. $1_{j,i}^{obj} = 1$ represents i^{th} boundary box in i cell and ∂_{record} **increase the weight for the loss.**

The confidence loss, which measures how object-like the box is, is as follows if an object is found inside:

$$Confidence\ loss = \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{obj} \ (w_j(i) - \hat{h}_j(i))^2 \quad (3)$$

The loss of confidence in the occurrence that an object is not found in the box can be calculated using the following equation:

$$Confidence\ loss = \partial_{noobj} \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{noobj} \ (w_j(i) - \hat{h}_j(i))^2 \quad (4)$$

The sum of the localization, confidence, and classification losses determines the final loss using the following equation:

$$Final\ Loss = \partial_{record} \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{obj} [(p_j(i) - \hat{p}_j(i))^2 + (y_j(i) - \hat{y}_j(i))^2] + \partial_{record} \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{obj} [(h_j(i) - \hat{h}_j(i))^2 + (l(i) - \hat{l}_j(i))^2] + \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{obj} \ (w_j(i) - \hat{h}_j(i))^2 + \partial_{noobj} \sum_{j=0}^{m^2} \sum_{i=0}^{A} 1_{j,i}^{noobj} \ (w_j(i) - \hat{h}_j(i))^2$$

(5)

The YOLO model is widely used for real-time object detection due to its speed and accuracy. Various adaptations have improved their efficiency in applications such as garbage classification using a Variational Autoencoder (VAE) [20], robotic vision [21], medical face mask detection [22], and traffic monitoring [23]. Compact versions like Tiny YOLOv2 and Mini-YOLOv3 have further optimized detection performance for embedded systems [24, 26].

For fire detection, Convolutional Neural Network (CNN)- and Internet of Things (IoT)-integrated YOLOv5 models have enhanced real-time monitoring while minimizing false alarms [27]. However, earlier models faced challenges in detecting small fires and adapting to environmental variations due to dataset limitations and computational constraints [28–32]. Recent advancements in YOLOv5 and YOLOv8, incorporating Distance-IoU Non-Maximum Suppression (DIoU-NMS) for improved bounding box suppression [33], defogging techniques [34], and enhanced feature extraction with Scaled IoU (SIoU) and Convolutional Block Attention Module (CBAM) [35], have significantly boosted accuracy and efficiency. FireYOLO, combined with

Real-Enhanced Super-Resolution Generative Adversarial Network (Real-ESRGAN) has also demonstrated improved remote fire detection by enhancing target recognition [36].
This study builds on previous research by integrating YOLOv8 for industrial fire detection. Employing fine-tuning and dataset augmentation enhances detection performance, ensuring improved object localization and reduced false positives, making it a reliable choice for real-time fire monitoring systems.

This study enhances previous research by integrating YOLOv8 for fire detection in industrial settings. It improves model performance through fine-tuning and dataset augmentation. Its advanced object localization and reduced false positives make it an effective choice for real-time fire monitoring systems.

3. Dataset Description

The team for the 2018 NASA Space Apps Challenge created the data utilized in this study from a variety of sources, including https://www.kaggle.com/phylake1337/fire-dataset, The NASA Space data is divided into two folders: fire images, which contains 755 outdoor fire shots, some of which contain a lot of smoke, and non-fire images, which contains 244 nature photos (including forests, trees, grass, rivers, people, foggy forests, lakes, animals, roads, and waterfalls). More data is gathered from https://github.com/sulenn/fire-dataset. There are 3203 different fire pictures about candles, forests, accidents, experiments, and so on. Pictures are compressed into 7 packages. Because the size of the pictures is different, some packages only have a few pictures. These pictures come from the GitHub Repository and a few fire videos.
Figure 2 displays examples of images with and without fire.

FIGURE 2. Sample images with and without fire.

4. Proposed Model

The suggested method is intended to address some of the shortcomings of earlier studies in the detection of fire and offer increased accuracy, real-time detection, cost-effectiveness, and precision. This paper presented a You Only Look Once Version 8 (YOLOv8) framework-based model for industrial fire detection. The three main phases of the suggested model are feature selection, data pre-processing, and outcome evaluation using a range of metrics as illustrated in FIGURE 3. During the data preprocessing step, the images are resized and enhanced, noise is reduced, and videos and images are labeled with bounding boxes surrounding the fires. The YOLOv8 model is the recommended option for feature selection due to its speed and accuracy. Metrics such as accuracy, precision, and recall are employed to assess the performance of the proposed model. Moreover, the proposed model is trained in a real-time system that can process real-time video streams. When a fire is detected, the building's fire alarm should go on to alert people and tell them to evacuate.

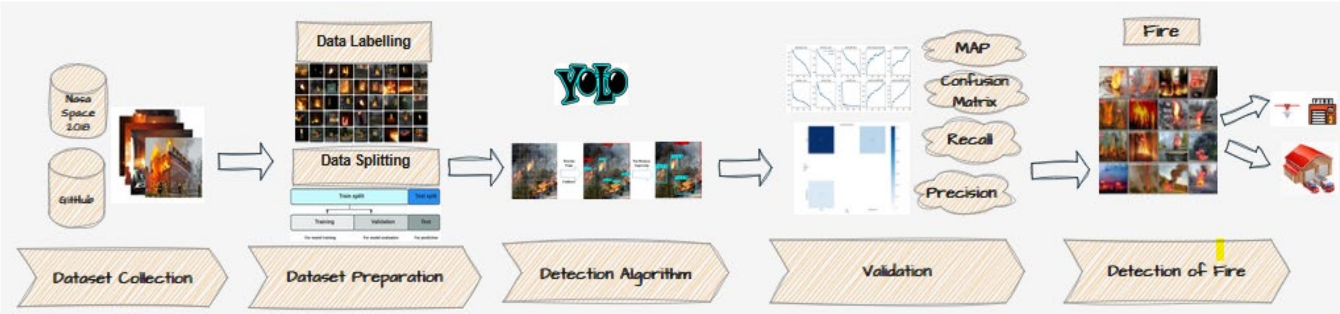

FIGURE 3 . Fire detection model.

Using live camera feeds or previously recorded video files, the fire detection algorithm employs computer vision to identify fires in real time. It employs a pre-trained Yolov8 object detection model on a sizable dataset of fire and non-fire photos, as demonstrated in Algorithm 1. It receives a dataset of video frames as input and output classes related to fire, as well as objects that are detected. Each video frame is iterated through by the algorithm, which preprocesses each one before sending it to the Yolov8 model for object detection. When an algorithm detects a class related to fire, it sounds like an alarm and notifies the appropriate authorities. The algorithm then saves the resultant video, highlighting the objects it has found. Real-time fire detection is made possible by the Fire Detection Algorithm, which also facilitates prompt and efficient responses to possible fire threats. There are multiple steps involved:

1. A pre-recorded video file or a live camera is configured as the video input source.
2. The video is first captured, and every frame is played backward and forward.
3. Every frame is subjected to image pre-processing techniques, after which the Yolov8 model receives the pre-processed frame for object detection.
4. Classes related to fire are examined in the objects detected.
5. When a class related to fire is found, an alarm is set off, and the appropriate authorities are alerted.
6. The output video is saved with the identified objects highlighted, and the video capture process is terminated.

Algorithm 1: Fire Detection Algorithm

Input: recorded video
Output: Detected object
1-Load the pre-trained Yolov8 object detection model and set up the necessary configurations.
2-set up the video input score.
3- Start the video capture process.
4- Loop through every frame in the video.
  a) Apply image pre-processing methods like resizing and normalization to the current frame.
  b) Pass the pre-processed frame to the Yolov8 model for object detection.
  c) Check the detected objects for fire-related classes.
  d) If a fire-related class is detected, an alarm is set off and the appropriate authorities are alerted.
5- Start the process of recording a video.
While True:
- Read current frame from video input source.
- Apply image pre-processing techniques.

6- Stop the video capture process.
7- The output video, which highlights the objects that were detected, is saved.

4.1 Data preprocessing

Before starting data preprocessing, data gathering is necessary. To complete this step, a sizable dataset of both fire and non-fire images and videos must be gathered. This can be accomplished by gathering pictures and videos from open sources, like news websites and social media, or by filming with specific cameras or sensors. The dataset must be carefully selected, duplicates must be eliminated, and either manually or automatically labeled methods must be used to indicate whether or not fires have occurred. To avoid the model being skewed toward one type of image, it is crucial to make sure the dataset is balanced with an equal number of images showing fire and images showing no fire. In data preprocessing, photos and videos—both fire and non-fire—for the fire detection system's training and testing are prepared in the dataset. Using a tool for Labeling the photos and videos with bounding boxes surrounding the fires is what this entails. Next, to guarantee that both training and testing sets are representative of the entire dataset, the labeled data are divided into sets. It might also be required to perform additional pre-processing operations like data normalization or resizing. A balanced, sizable dataset that performs well in terms of generalization to new data is the aim.

4.2 Model Selection

Roboflow online framework is used for model selection steps. The computer vision framework Roboflow simplifies the development and application of computer vision models. It provides resources for training, labeling, augmentation, and implementation of data. Roboflow is used by developers, researchers, and organizations to build a variety of computer vision applications, including image classification, object identification, and semantic segmentation. With Roboflow, you can upload your datasets, enhance their labeling, add missing data, and create new datasets based on the ones that are already available. FIGURE 4 shows custom labeling in a corrupted file that did not have the label assigned by the original dataset.

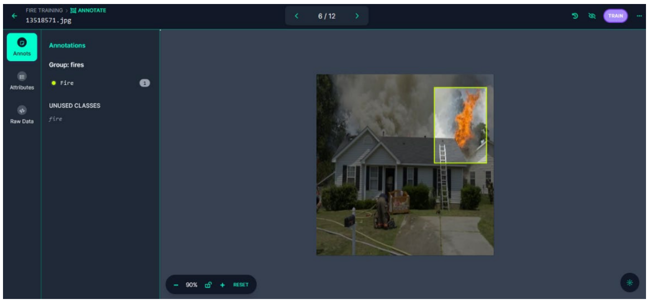

FIGURE 4 . Labelling dataset images by using Roboflow.

The YOLOv8 model [19], on the other hand, is the latest version of YOLO models, built in 2023 by Ultralytics. It is the most accurate and fastest object detection algorithm to date. It contains a set of distinguished features like:

- CSPDarkNet53, which is a brand-new backbone network that replaces DarkNet53 and is more precise and efficient than YOLOv5.
- PANet, which enhances object detection precision by more effectively combining features from various scales.
- GIoU loss function, which is in comparison to the IoU loss function used earlier.

Finally, we can export our custom-generated dataset in the format we need, which in our case is the YOLOv8 model, but we can see that we have plenty of other models, which makes this tool easily compatible with the main state-of-the-art object detection models, as shown in FIGURE 5.

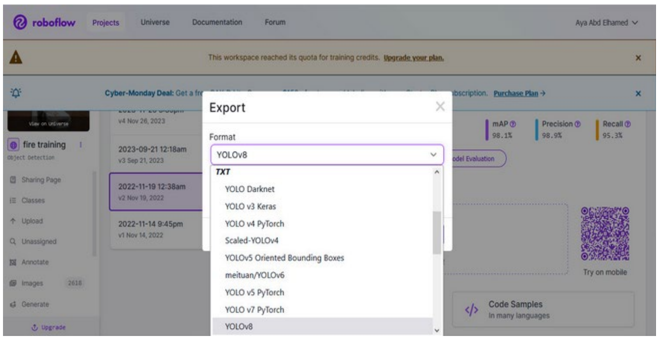

FIGURE 5. Exporting custom generated dataset in YOLOv8 version (Roboflow).

4.3 Deployment Step

Using live video streams from cameras, the trained model is implemented in a real-time system. A robust computer or server equipped with a GPU is necessary for real-time processing of the video feeds. When a fire is detected, the system should be able to read video frames from a camera or video stream, process them using the trained model, and produce alerts. A threshold value establishes the minimal confidence level required for the model to identify a fire; detections below this threshold value are rejected as false positives. This is one way to handle false positives.

4.4 Integration Step

The process of integration entails connecting the fire detection system to other systems, including emergency response, sprinkler, and fire alarm systems. The building's fire alarm system ought to sound when a fire is discovered, warning occupants to leave. To put out the fire, the sprinkler system can also be turned on. To provide a prompt and efficient response, emergency response systems can also be alerted to vital information on the location and intensity of the fire. These systems must be integrated properly to prevent false alarms and guarantee a quick, effective fire response. It is important to conduct testing and validation to make sure the systems function as a unit.

In addition, to maintain the deployed fire detection system's efficacy over time, maintenance is required. This includes adding new data to the model, testing the system on a regular basis, and keeping the system's hardware and software up to date. Frequent maintenance enhances general safety by lowering the possibility of false alarms.

5. Experimental Results and Discussion

The performance of the experimental results is evaluated using a variety of metrics, including recall, F-Measure, accuracy, and precision.

The metrics have the following definitions:

Confusion matrix, a tabular representation of an algorithm's performance. It displays the quantity of true positives (TP, FP) and false negatives (TN, FN) for each class.

Precision is the percentage of the objects being correctly identified as belonging to a certain class. Precision is the proportion of objects that are accurately classified as members of a particular class. Precision in terms of positive findings is the ratio of correctly anticipated observations to all positive finds that were predicted. The ratio of true positive predictions to all positive predictions is known as precision. This can be computed using Equation (6).

Recall the percentage of objects that are identified as part of a certain class. The recall is the proportion of true positive predictions over all actual positives. This can be computed using Equation (7).

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

$$recall = \frac{TP}{TP+FN} \tag{7}$$

where True Positive is represented by TP. False Positive is referred to as FP, and True Negative as TN.

MAP (Mean Average Precision), the most used performance evaluation metric in the object detection model, it is the average of the Average Precision metric over all the classes in a certain model. This metric has two main sections: mAP(5) and mAP(5-95). The "(5)" means that it is computed at a threshold of 0.5(or 50%) Intersection-over-Union (IoU). The "(5-95)" means that it is calculated across a range of IoU thresholds, from 0.5 to 0.95 with a step size of 0.05. These metrics have been used to evaluate the overall performance of our models, but we have also observed the training time, CPU/GPU usage, and adaptability to a certain environment/platform.

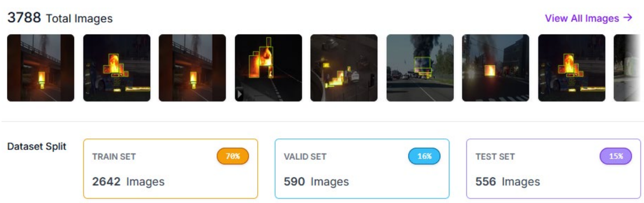

FIGURE 6. Dataset split

The YOLOv8 model was trained using transfer learning, initializing with pre-trained COCO dataset weights and fine-tuning on the collected dataset. Training was conducted for 30 Epochs with an initial learning rate of 0.01 and a batch size of 16. The dataset consists of 3,788 images, split into 70% for training, 15% for validation, and 15% for testing, as illustrated in Figure 6. Fire annotations follow the YOLOv8 format.

Each image underwent preprocessing, including resizing to 640×640 and auto-orientation of pixel data (with EXIF-orientation stripping), and no image augmentation was applied. The proposed model was implemented, trained, and validated on a GPU platform. An example of fire detection is shown in Figure 7.

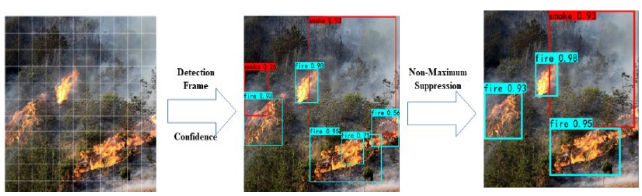

FIGURE 7 . Fire Detection Examples

The epoch versus loss of the model on train and validation datasets is shown in FIGURE 8.

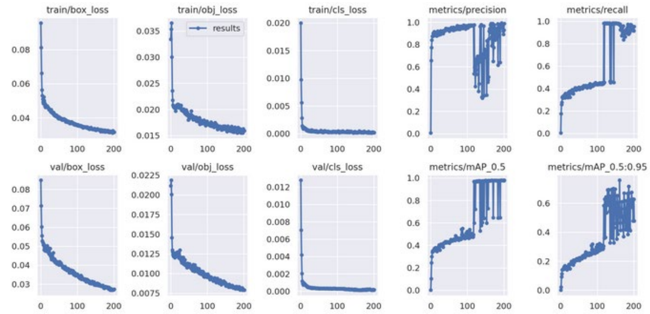

FIGURE 8 . Train & Validate Metrics

Figure 8 presents the training and validation results of YOLOv8, highlighting key loss components: train/box_loss, train/cls_loss, and train/dfl_loss represent bounding box loss, classification loss, and distribution focal loss during training, while val/box_loss, val/cls_loss, and val/dfl_loss indicate the same during validation. Evaluation metrics, including mAP50 and recall, measure detection performance, with mAP50 reflecting mean average precision at an IoU of 0.50. Box and Objectness assess the accuracy of bounding box placement, while Classification evaluates object detection performance. The results demonstrate effective learning, as both training and validation losses decrease consistently. Precision and recall improved over time, with mAP50 and mAP50-95 steadily increasing, confirming enhanced detection accuracy. This indicates that YOLOv8 successfully generalizes fire detection applications.

TABLE I . The evaluation metrics for different models

Model	Precision	Recall
YOLOv3	92.09%	88.89%
YOLOv4	93.57%	87.2%
YOLOv5	85.2%	25.07%
YOLOv6	52.8%	70.6%
YOLOv7	78.9%	67.7%
YOLOv8	**98.9%**	**95.3%**

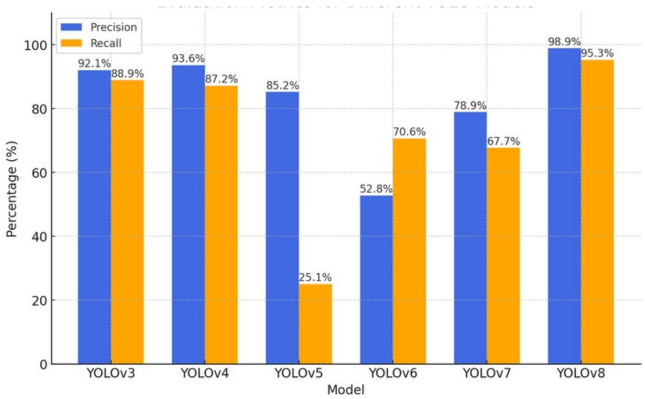

Figure 9. Evaluation Metrics for Different YOLO Models.

The results in Table I and Figure 9 provide a comparative analysis of YOLO models from YOLOv3 to YOLOv8 using the same dataset. The results show that YOLOv8 outperforms its predecessors in precision and recall, making it highly effective for real-world fire detection. The proposed model achieves a high precision rate of 98.9% in distinguishing fire from non-fire instances. This improvement is attributed to YOLOv8's enhanced feature extraction, optimized training strategies, and architectural advancements, such as improved anchor box predictions and adaptive spatial feature fusion. Unlike earlier YOLO versions and CNN-based models, YOLOv8 excels in detecting small fires and adapting to varying conditions, proving to be a reliable choice for real-time fire detection applications.

TABLE II . Comparison of various YOLOv8 models and their evaluation metrics.

Model	Precision	Recall	MAP50	MAP50-95	Time (Hours)
YOLOv8n	91.7%	86.5%	93.5%	58.5%	0.779
YOLOv8s	94.3%	**91.9%**	96.4%	67.2%	0.812
YOLOv8m	**96.2%**	86.6%	**95.1%**	63.5%	1.452
YOLOv8l	90.5%	82.3%	90.3%	56.5%	2.167
YOLOv8x	91.6%	82.8%	90.4%	56.8%	3.675

The performance of the five YOLOv8 models, assessed using four metrics—precision, recall, mean average precision at IoU 0.50 (mAP50), and mean average precision at IoU 0.50 to 0.95 (mAP50-95)—along with training time. The results in Table II and Figure 10 highlight the performance trade-offs among different YOLOv8 variants in fire detection tasks. YOLOv8m achieved the highest precision (96.2%) and a strong MAP50 (95.1%), likely due to its balanced architecture, which provides a good mix of accuracy and computational efficiency. YOLOv8s demonstrated the best recall (91.9%) and the highest MAP50-95 (67.2%), suggesting its ability to detect a wide range of fire instances, benefiting from an optimized feature extraction process and fine-tuned hyperparameters. YOLOv8n, with its lightweight design, exhibited competitive precision (91.7%) and recall (86.5%) while maintaining the shortest training time (0.779 hours), making it suitable for real-time applications with limited resources. On the other hand, YOLOv8l and YOLOv8x required significantly longer training times (2.167 and 3.675 hours, respectively) but did not outperform YOLOv8m or YOLOv8s in terms of precision and recall. This can be attributed to diminishing returns in accuracy gains despite increased model complexity. The results justify the selection of YOLOv8m and YOLOv8s as optimal choices, balancing detection accuracy and computational efficiency for real-world fire detection applications.

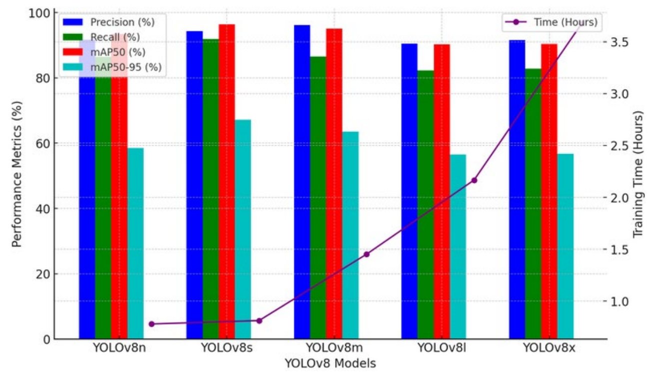

Figure 10 . Comparison of various YOLOv8 models and their evaluation metrics.

TABLE III . The proposed model vs. The state-of-the-art models

Ref.	Model	Precision	Recall	maP	System
[37]	YOLO V8	97.5%	95.7%	96.2%	Fire / smoke
[38]	YOLO V5	94.99%	78.28%	85.8%	Smoke
[39]	YOLO V6	93.48%	28.29%	90%	Fire / smoke
[40]	YOLO V2	97%	97%	95.4%	Fire / smoke
Proposed model	**YOLO V8**	**98.9%**	**95.3%**	**98.1%**	**Fire /non_fire**

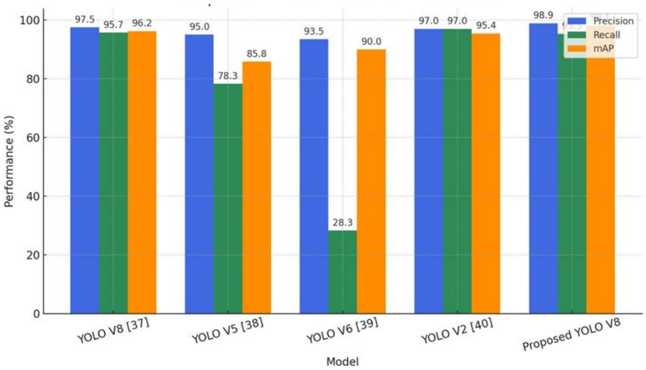

FIGURE 11 . Comparison of YOLO Models for Fire Detection.

The three systems' performance comparison is displayed in TABLE III and Figure 11. The suggested model demonstrated its efficacy in identifying fires in real-world scenarios by outperforming the current models in terms of maP, precision and recall. Although the authors in [37] and our model used the same version of YOLO, our suggested model has a high precision rate of 98.9% in detecting fire and non-fire. When compared to other deep learning-based fire detection systems and previous versions of YOLO, we believe that YOLO v8's increased accuracy and speed detection for our system's superior performance.

6. Conclusion

This paper presented an improved method for industrial fire detection based on the YOLOv8 algorithm which makes use of deep learning capabilities to detect fire-specific characteristics instantly in real time. In comparison to more conventional fire detection techniques, the fire detection approach may increase fire detection accuracy, lower false alarms, and be more affordable. It can also be expanded to detect other interesting objects in smart cities, like flooding or gas leaks. This study presented a You Only Look Once Version 8 (YOLOv8) framework-based model for industrial fire detection. The three main phases of the suggested model are feature selection, data pre-processing, and outcome evaluation using a range of metrics. During the data pre-processing step, images are resized, enhanced, noise is reduced, and videos and images are labeled with bounding boxes surrounding the fires. The YOLOv8 model is the recommended option for feature selection due to its speed and accuracy. The accuracy, precision, and recall of the suggested model are just a few of the metrics used to evaluate its performance. Moreover, the proposed model is trained on a real-time system that can process real-time video streams. The building's fire alarm system should sound when a fire is discovered to warn people and direct them to leave. The results of the experiment demonstrate that the suggested model achieved 98.1% accuracy, 98.9% precision, 95.3% recall, and 98.1% mAP. Finally, using the same dataset, the suggested model is compared to current techniques. In future we will continue further research and experimentation with the most recent object detection models, such as YOLOv11, which have been released recently.

Author's contributions: All authors contributed equally to each part of the article.

Conflicts of Interest: All authors declare no conflict of interest.

Data Availability: The datasets used during the current study are available from the corresponding author upon reasonable request.

Reference

1. Chen, X.; Ling, J.; Wang, S.; Yang, Y.; Luo, L.; Yan, Y. Ship detection from coastal surveillance videos via an ensemble Canny-Gaussian-morphology framework. J. Navig. 2021, 74, 1252–1266. [CrossRef]
2. Marbach, G.; Loepfe, M.; Brupbacher, T. An image processing technique for fire detection in video images. Fire Saf. J. 2006,41, 285–289. [CrossRef]
3. Foggia, P.; Saggese, A.; Vento, M. Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion. IEEE Trans. Circuits Syst. Video Technol. 2015, 25, 1545–1556. [CrossRef]
4. Mueller, M.; Karasev, P.; Kolesov, I.; Tannenbaum, A. Optical flow estimation for flame detection in videos. IEEE Trans. Image Process. 2013, 22, 2786–2797. [CrossRef] [PubMed]
5. Muhammad, K.; Ahmad, J.; Mehmood, I.; Rho, S.; Baik, S.W. Convolutional neural networks-based fire detection in surveillance videos. IEEE Access 2018, 6, 18174–18183. [CrossRef]
6. Dunnings, A.J.; Breckon, T.P. Experimentally defined convolutional neural network architecture variants for non-temporal real-time fire detection. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018.
7. Mei, X.; Han, D.; Saeed, N.; Wu, H.; Chang, C.-C.; Han, B.; Ma, T.; Xian, J. Trajectory Optimization of Autonomous Surface Vehicles with Outliers for Underwater Target Localization. Remote Sens. 2022, 14, 4343. [CrossRef]
8. Zhao, M.; Hu, C.; Wei, F.; Wang, K.; Wang, C.; Jiang, Y. Real-time underwater image recognition with FPGA embedded system for convolutional neural network. Sensors 2019, 19, 350. [CrossRef]
9. Chen, T.H.; Wu, P.H.; Chiou, Y.C. An early fire-detection method based on image processing. In Proceedings of the International Conference on Image Processing (ICIP), Singapore, 24–27 October 2004; pp. 1707–1710. [Google Scholar]

10. Celik, T.; Demirel, H.; Ozkaramanli, H.; Uyguroglu, M. Fire detection using statistical color model in video sequences. J. Vis. Commun. Image Represent. **2007**, 18, 176–185. [Google Scholar] [CrossRef]
11. Wang, T.; Shi, L.; Hou, X.; Yuan, P.; Bu, L. A new fire detection method based on flame color dispersion and similarity in consecutive frames. In Proceedings of the 2017 Chinese Automation Congress (CAC), Jinan, China, 20–22 October 2017; pp. 151–156. [Google Scholar]
12. Borges, P.V.K.; Izquierdo, E. A probabilistic approach for vision-based fire detection in videos. IEEE Trans. Circuits Syst. Video Technol. **2010**, 20, 721–731. [Google Scholar] [CrossRef]
13. Mueller, M.; Karasev, P.; Kolesov, I.; Tannenbaum, A. Optical flow estimation for flame detection in videos. IEEE Trans. Image Process. **2013**, 22, 2786–2797. [Google Scholar] [CrossRef] [PubMed]
14. Foggia, P.; Saggese, A.; Vento, M. Real-time fire detection for video surveillance applications using a combination of experts based on color, shape, and motion. IEEE Trans. Circuits Syst. Video Technol. **2015**, 25,
15. Wang, X., Y. Li and Z. Li. 2020 Research on flame detection algorithm based on multi-feature fusion. 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), IEEE. https://doi.org/10.1109/ITNEC48623.2020.9084825
16. Ghosh, R., Kumar, A. A hybrid deep learning model by combining convolutional neural network and recurrent neural network to detect forest fire. *Multimed Tools Appl* **81**, 38643–38660 (2022). https://doi.org/10.1007/s11042-022-13068-8
17. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified,Real-Time Object Detection", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, (2016), pp. 779-788
18. Jocher, G. 2020. "YOLOv5 by Ultralytics (Version 7.0)." Computer software. doi:10.5281/zenodo.3908559.
19. Mupparaju, S., Thotakura, S., & Venkata RamiReddy, C. (Year). *A Review on YOLOv8 and its Advancements*. School of Computer Science and Engineering, VIT-AP University, Amaravati, India.
20. Ye A, Pang B, Jin Y, Cui J (2020) A YOLO-based neural network with VAE for intelligent garbage detection and classification. In 2020 3rd international conference on algorithms computing and artificial intelligence pp 1-7
21. Zheng Y, Ge J (2021) Binocular intelligent following robot based on YOLO-LITE. In MATEC web of conferences (Vol 336 p 03002) EDP sciences
22. Loey M, Manogaran G, Taha MHN, Khalifa NEM (2021) Fighting against COVID-19: a novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection. Sustain Cities Soc 65:102600
23. Zhang X, Qiu Z, Huang P, Hu J, Luo J (2018) Application research of YOLO v2 combined with color identification. In 2018 international conference on cyber-enabled distributed computing and knowledge discovery (CyberC) pp 138-1383
24. Cao Z, Liao T, Song W, Chen Z, Li C (2021) Detecting the shuttlecock for a badminton robot: a YOLO based approach. Expert Syst Appl 164:113833. https://doi.org/10.1016/j.eswa.2020.113833
25. Chen B, Miao X (2020) Distribution line pole detection and counting based on YOLO using UAVinspection line video. J Electr Eng Technol 15(1):441–448. https://doi.org/10.1007/s42835-019-00230-w
26. Mao QC, Sun HM, Liu YB, Jia RS (2019) Mini-YOLOv3: real-time object detector for embedded applications. IEEE Access 7:133529–133538
27. Abdusalomov AB et al (2023) An improved forest fire detection method based on the detectron2 model and a deep learning approach. Sensors 23(3):1512
28. Li P, Zhao W (2020) Image fire detection algorithms based on convolutional neural networks. Case Stud Therm Eng 19:100625
29. Muhammad K, Ahmad J, Mehmood I, Rho S, Baik SW (2018) Convolutional neural networks-based fire detection in surveillance videos. IEEE Access 6:18174–18183
30. Pan H, Badawi D, Cetin AE (2020) Computationally efficient wildfire detection method using a deep convolutional network pruned via Fourier analysis. Sensors 20:2891
31. Li T, Zhao E, Zhang J, Hu C (2019) Detection of wildfire smoke image based on a densely dilated convolutional network. Electronics 8:1131

32. Kim B, Lee J (2019) A video-based fire detection using deep learning models. Appl Sci 9:2862
33. Wu, Z., R. Xue, and H. Li. 2022. Real-time video fire detection via modified YOLOv5 network model. *Fire Technology* 58 (4): 2377–2403. https://doi.org/10.1007/s10694-022-01260-z.
34. Huang, J., Z. He, Y. Guan, and H. Zhang. 2023. Real-time forest fire detection by ensemble lightweight YOLOX-L and defogging method. *Sensors* 23 (4): 1894. https://doi.org/10.3390/s23041894.
35. Xue, Q., H. Lin, and F. Wang. 2022. FCDM: An improved forest fire classification and detection model based on YOLOv5. *Forests* 13 (12): 2129. https://doi.org/10.3390/f13122129
36. Zheng, H., S. Dembélé, Y. Wu, Y. Liu, H. Chen, and Q. Zhang. 2023. A lightweight algorithm capable of accurately identifying forest fires from remotely sensed images. *Frontiers in Forests and Global Change* 6: 36. https://doi.org/10.3389/ffgc.2023.1134942
37. Sridhar, P.; Thangavel, S.K.; Parameswaran, L.; Oruganti VR, M. Fire Sensor and Surveillance Camera-Based GTCNN for Fire Detection System. IEEE Sens. J. **2023**, 23, 7626–7633. [CrossRef]
38. Talaat, Fatma M., and Hanaa ZainEldin. "An improved fire detection approach based on YOLO-v8 for smart cities." Neural Computing and Applications 35.28 (2023): 20939-20954.
39. Wang, Zhong, et al. "A smoke detection model based on improved YOLOv5." Mathematics 10.7 (2022): 1190.
40. Norkobil Saydirasulovich, Saydirasulov, et al. "A YOLOv6-based improved fire detection approach for smart city environments." Sensors 23.6 (2023): 3161.