Neural Computing and Applications (2023) 35:20939–20954
https://doi.org/10.1007/s00521-023-08809-1

ORIGINAL ARTICLE

An improved fire detection approach based on YOLO-v8 for smart cities

Fatma M. Talaat[1] · Hanaa ZainEldin[2]

Received: 28 March 2023 / Accepted: 28 June 2023 / Published online: 28 July 2023

Abstract

Fires in smart cities can have devastating consequences, causing damage to property, and endangering the lives of citizens. Traditional fire detection methods have limitations in terms of accuracy and speed, making it challenging to detect fires in real time. This paper proposes an improved fire detection approach for smart cities based on the YOLOv8 algorithm, called the smart fire detection system (SFDS), which leverages the strengths of deep learning to detect fire-specific features in real time. The SFDS approach has the potential to improve the accuracy of fire detection, reduce false alarms, and be cost-effective compared to traditional fire detection methods. It can also be extended to detect other objects of interest in smart cities, such as gas leaks or flooding. The proposed framework for a smart city consists of four primary layers: (i) Application layer, (ii) Fog layer, (iii) Cloud layer, and (iv) IoT layer. The proposed algorithm utilizes Fog and Cloud computing, along with the IoT layer, to collect and process data in real time, enabling faster response times and reducing the risk of damage to property and human life. The SFDS achieved state-of-the-art performance in terms of both precision and recall, with a high precision rate of 97.1% for all classes. The proposed approach has several potential applications, including fire safety management in public areas, forest fire monitoring, and intelligent security systems.

Keywords Smart city · Fire detection · YOLOv8 · Deep learning

1 Introduction

The manner we think about urbanization, sustainability, and safety is being completely transformed by smart cities. As the world moves toward smart cities, it becomes increasingly important to ensure the safety of citizens and their properties [1]. One of the most dangerous and life-threatening catastrophes is a fire, which may seriously harm both property and people. Fire accidents pose a significant threat to smart cities as they can cause significant damage to infrastructure, lead to loss of life, and disrupt the smooth functioning of the city [2]. Therefore, it is crucial to have an early fire detection system that is effective and reliable. Early fire detection is now a top priority in smart cities due to the rising urbanization and increased awareness of the value of safety. Early fire detection and action can reduce property damage while also saving lives. However, this endeavor necessitates handling difficulties including the unpredictable nature of fire, the requirement for ongoing observation, and the enormous amounts of data produced by smart cities [3].

To detect fires early, researchers and engineers have created vision-based fire detectors (VFDs), as well as fire sensors that are sound sensitive, flame sensitive, temperature sensitive, gas sensitive, or solid sensitive [4]. Sensors pick up on the chemical characteristics of smoke, setting off an alarm. This strategy, nevertheless, might cause erroneous warnings. Once the smoke is close enough proximate to the sensors to trigger them, the alarm will not sound. Those monitoring systems, which were developed as parts of conventional alarm systems, sensed the flame's smoke and temperature as well as other flame-related

✉ Fatma M. Talaat
fatma.nada@ai.kfs.edu.eg

Hanaa ZainEldin
eng.hanaa2011@gmail.com

[1] Faculty of Artificial Intelligence, Kafrelsheikh University, Kafrelsheikh, Egypt

[2] Computers and Control Systems Engineering Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt

characteristics. A sensor-based detection system [5] may not be viable in some situations, such as those involving wide coverage areas, untamed (forest areas), or high temperatures, as it will provide a lot of false alerts [6].

Traditionally, fire detection systems have depended on temperature, gases, and smoke sensors, which have been established to be successful for small fires but ineffective for larger fires that can grow rapidly, devour the entire region, and have disastrous effects. The implementation of deep learning techniques to improve the detection of fires in real time has been encouraged by the advent of IoT-enabled smart cities [4, 7].

Deep learning techniques for early fire detection have been the subject of several prior investigations. For instance, a fire detection system (FFireNet) was proposed by [8] using the MobileNetV2 model to classify forest fires. Additionally, Mukhiddinov et al. [9] proposed an early wildfire smoke detection system based on improved YOLOv5 photographs taken by unmanned aerial vehicles (UAVs). A fire detection technique based on an improved YOLO V4 is proposed in [10]. According to the experimental results, the proposed technology can be applied effectively to defend smart cities and to keep track of fires in urban areas.

Convolutional neural networks (CNNs) have shown remarkable performance in image recognition tasks, including object detection. The You Only Look Once (YOLO) algorithm is one such CNN-based object detection framework that has been widely used in computer vision applications. The latest YOLO v8 version has shown significant improvements in accuracy and speed, making it a viable option for real-time fire detection in smart cities.

This paper presented an early fire detection system based on YOLO v8 for smart cities. The system architecture described, and the experimental results provided exhibit its effectiveness in detecting fires in real-world scenarios. The proposed approach compares with existing fire detection systems and highlights the advantages of our proposed system.

The main contribution of this paper is:

- Proposing a smart city framework which is composed of four main layers which are: (i) Application layer, (ii) IoT layer, (iii) Fog layer, and (iv) Cloud layer. This work focuses on six main applications: (i) Smart Government, (ii) Smart Street, (iii) Smart Hospitals, (iv) Smart Home, (v) Smart Traffic System, and (vi) Green Application.
- Proposing a YOLOv8-based improved fire detection approach for smart cities.
- Improving accuracy: The proposed approach may improve the accuracy of fire detection in smart cities compared to traditional methods. This can be achieved by leveraging the strengths of deep learning algorithms such as YOLOv8 to learn and detect fire-specific features that may be difficult to identify using traditional image processing methods.
- Real-time detection: The YOLOv8 algorithm is known for its speed and ability to perform object detection in real time. This makes the proposed approach well-suited for smart city applications where quick and timely detection of fires is critical.
- Versatility: The proposed approach can be easily adapted to detect other objects of interest in smart cities, making it a versatile tool for various applications beyond fire detection. For instance, it can be used to detect other safety hazards, such as gas leaks or flooding, or to monitor traffic and pedestrian flow in crowded areas.
- Reduced false alarms: By using deep learning to learn fire-specific features, the proposed approach may be able to reduce false alarms that are common in traditional fire detection methods. This can help to avoid unnecessary emergency responses and reduce costs associated with false alarms.
- Cost-effective: The proposed approach may be cost-effective compared to traditional fire detection methods as it can be implemented using low-cost cameras and hardware, reducing the need for expensive fire detection systems.
- Large dataset: Unlike other methods that use small number of datasets, a large dataset containing fire, smoke, and normal scenes is used. The dataset has real-world images and videos collected from various sources. The dataset has a diverse range of fire scenarios, including indoor and outdoor fires, small and large fires, and low-light and high-light conditions. A deep CNN gathers essential data from big datasets to produce precise predictions and reduce overfitting.

The following is how the remaining work is structured. Section 2 presents some of the most recent research in the field of AI in smart cities and 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 Related work

Recently, there has been a growing interest in using deep learning-based approaches for fire detection in smart cities. Some studies have proposed hybrid approaches that combine multiple deep learning algorithms for fire detection. For instance, Al-Turjman et al. [11] proposed a hybrid approach that combined CNN and recurrent neural network (RNN) for fire detection in smart cities. The proposed

approach achieved high accuracy and low false alarm rates. Another study by Huang et al. [12] proposed a YOLOv3-based approach for fire detection in outdoor scenes. This approach evaluated on a dataset of real-world images and achieved an accuracy of 92.8%.

Other works have focused on using multiple deep learning models for fire detection. For example, the study by Jia et al. [13] proposed a multi-model approach that combined a CNN and a long short-term memory (LSTM) network for fire detection. The approach was evaluated on a dataset of video frames and achieved an accuracy of 96.3%. Another notable work is the study by Wang et al. [14], which proposed a deep learning-based approach for fire detection in surveillance videos. The approach was based on the YOLOv2 algorithm and achieved an accuracy of 93.6%. More recently, the YOLOv3 algorithm has been used for fire detection in smart cities.

Another deep learning-based approach for fire detection is the convolutional neural network (CNN). CNNs are a class of deep neural networks that are particularly well suited for image processing tasks. CNN-based fire detection systems have been proposed by several researchers. For example, He et al. [15] proposed a CNN-based fire detection system that achieved high accuracy in detecting fires in videos. Some of the notable works in this area are discussed below. One of the early works in this area is the study by Shen et al. [16], which proposed a deep learning-based approach for fire detection using a convolutional neural network (CNN). The approach was evaluated on a dataset of indoor and outdoor fire images and achieved an accuracy of 91%.

Ba et al. [17] proposed a novel Convolutional Neural Network (CNN) model called Smoke Net to improve feature representations for visual classifications by incorporating spatial and flow attention mechanisms. Meanwhile, Luo et al. [18] proposed a CNN-based approach for identifying flames, which utilized the kinetic characteristics of smoke to separate potential candidates into two groups based on dynamic frame references from the backdrop and the foreground. A CNN with five convolutional layers plus three fully linked layers was then utilized to automatically retrieve the highlights of candidate pixels. In addition, deep convolutional segmentation networks have been developed for analyzing fire emergency scenes, which focused on identifying and classifying items in an image based on their construction information, such as color, relatively high intensity compared to their surroundings, various shifts in form and size, and the items' propensity to catch fire [19].

Researchers in [20] suggested earlier fire detection and warning systems for real-time monitoring of catastrophes caused by fire using cutting-edge techniques by combining the IoT with YOLOv5. The experimental findings demonstrate that some of the wrongly identified or unreported fires that YOLOv5 observed might be verified by IoT devices. This report is written down and forwarded to the fire station for additional confirmation. Authors in [21] embedded ML techniques operating on an energy-efficient device and a Video Surveillance Unit (VSU) for spotting and notifying the existence of forest fires. Timely fire detection is made possible by the ML models, which use audio samples and pictures as the corresponding inputs. The key finding is that while the two models' performances are equivalent, using them together following the suggested technique results in greater accuracy, precision, recall, and F1 score (96.15%, 92.30%, 100.00%, and 96.0%, respectively).

A transfer learning-based approach employing the previously trained InceptionResNetV2 network for classifying the Smoking and Non-Smoking imagery is proposed in [22]. On several performance measures, the effectiveness of the suggested method for predicting smoke and non-smoke was assessed and compared to existing CNN approaches. On an extensive and varied freshly constructed dataset, the suggested method correctly predicted the Smoking and Non-Smoking images with an accuracy of 96.87%, 97.32% precision, and 96.46% recall. To detect fire, an improved Inception-V3 has been proposed [23] on the fire and smoke images dataset. This model includes a new optimizing function that effectively lowers the cost of computation. When this work compared to other studies, the modified Inception v3-based model produced the best results in this research with the fewest false positives. Despite deploying additional sensors, [24] offers unique criteria termed fire alarm authenticity that uses the alert period of several smoke alarms to determine the location and severity of the fire location. The suggested criterion is used to construct an alert sequence identification algorithm, which is validated using simulations of actual fires and false alarms.

Previous studies have demonstrated the effectiveness of CNN models for fire detection, with reported maximum accuracy of 83.7% achieved by a unique picture fire detection system proposed in [25]. In addition, other studies have employed CNN techniques to improve the performance of image fire detection software [26–29]. However, these DL-based approaches often require large amounts of data for training, verification, and testing and suffer from spurious regressions and computational overhead due to the large datasets involved. To address these issues, a large dataset is compiled, which will soon be made accessible to the public. Table 1 presents the models that are commonly utilized for fire detection systems, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), YOLOv2, YOLOv3, YOLOv4, and YOLOv5.

Table 1 Comparative analysis of the prevalent models employed in the field

Model	Algorithm description	Pros	Cons
Convolutional neural networks (CNNs)	CNNs are a type of deep neural network that are particularly well suited for image processing tasks. They have been used for fire detection by training on images and identifying features that are common to fires, such as color and texture	CNNs are relatively simple to implement and can achieve high accuracy when trained on large datasets They can also identify fires in real time	CNNs may struggle with detecting fires in images with complex backgrounds or in low-light conditions They also require large amounts of data for training, which can be time-consuming and expensive
Recurrent neural networks (RNNs)	RNNs are a type of neural network that are designed to handle sequential data, such as time-series data. They have been used for fire detection by training on sequences of video frames and identifying patterns that indicate the presence of a fire	RNNs can capture temporal dependencies in the data, which can improve their accuracy in detecting fires in videos They are also flexible and can be used for a wide range of tasks, including speech recognition and natural language processing	RNNs can be computationally expensive and may struggle with long-term dependencies in the data They may also require large amounts of training data
YOLOv2	YOLOv2 is a deep learning algorithm that uses a single neural network to simultaneously predict bounding boxes and class probabilities for objects in an image. It has been used for fire detection by training on images and identifying features that are common to fires, such as color and texture	YOLOv2 can achieve high accuracy in object detection tasks and can process images in real time It is also relatively simple to implement	YOLOv2 may struggle with detecting fires in images with complex backgrounds or in low-light conditions It may also require large amounts of data for training
YOLOv3	YOLOv3 is an updated version of YOLOv2 that uses a more powerful neural network architecture and incorporates additional features, such as feature pyramid networks and residual connections. It has been used for fire detection in outdoor scenes	YOLOv3 can achieve high accuracy in object detection tasks and can process images in real time It is also more robust than YOLOv2 and can handle complex backgrounds and low-light conditions	YOLOv3 may require larger amounts of computational resources compared to YOLOv2
YOLOv4	OLOv4 is a further updated version of the YOLO algorithm that incorporates several new features, such as spatial attention modules and swish activation functions. It has been used for fire detection in smart cities	YOLOv4 can achieve high accuracy in object detection tasks and can process images in real time It is also more robust than previous versions of YOLO and can handle complex backgrounds and low-light conditions	YOLOv4 may require even larger amounts of computational resources compared to previous versions of YOLO
YOLOv5	YOLOv5 is a recent version of the YOLO algorithm that uses a lightweight network architecture and incorporates several new features, such as autoanchor optimization and multi-scale prediction. It has been used for fire detection in smart cities	Faster and more accurate than YOLOv4 Smaller and simpler architecture Three versions with different model sizes and complexities State-of-the-art performance on multiple object detection benchmarks	Still requires large amounts of data for training and may suffer from spurious regressions May have limitations in detecting small objects or objects with complex shapes

The table provides a comparative analysis of the prevalent models employed in the field of fire detection. However, some research gaps and limitations of these models are still present, such as:

- The CNNs may struggle with detecting fires in images with complex backgrounds or in low-light conditions.
- The RNNs can be computationally expensive and may struggle with long-term dependencies in the data, and they may also require large amounts of training data.
- YOLOv2 may struggle with detecting fires in images with complex backgrounds or in low-light conditions, and it may also require large amounts of data for training.
- YOLOv3 may require larger amounts of computational resources compared to YOLOv2.
- YOLOv4 may require even larger amounts of computational resources compared to previous versions of YOLO.

- YOLOv5 still requires large amounts of data for training and may suffer from spurious regressions, and it may have limitations in detecting small objects or objects with complex shapes.

Despite the advances in deep learning-based approaches for fire detection, there are still some challenges that need to be addressed. For example, there is a need for more diverse and larger datasets for training and testing these approaches. Additionally, the use of low-quality cameras or poor lighting conditions can affect the accuracy of fire detection algorithms.

In this paper, a YOLOv8-based approach for fire detection in smart cities is proposed. To the best of our knowledge, this is the first study to investigate the use of the YOLOv8 algorithm for fire detection in smart cities. The proposed approach is designed to address some of the limitations of previous studies and provide improved accuracy, real-time detection, versatility, reduced false alarms, and cost-effectiveness.

In the field of computer vision, the YOLO set of algorithms has achieved popularity. The popularity of YOLO is due to its high degree of accuracy while retaining a tiny model size. A wide spectrum of developers can use YOLO models since they can be trained on just one GPU. On edge hardware or in the cloud, machine learning experts may install it at a reasonable cost. The most recent and cutting-edge YOLO approach, YOLOv8, could be employed for applications including object identification, image categorization, and segmentation. Yolo v8 was produced by Ultralytics, who also produced the significant YOLOv5 model that defined the industry. Compared to YOLOv5, YOLOv8 has some architectural updates and enhancements [30].

The YOLOv8 model is anchor-free. It implies that rather than predicting an item's distance from a known anchor box, it estimates the center of the object explicitly. Anchor-free detection lowers the number of box predictions, which expedites Non-Maximum Suppression (NMS), a challenging post-processing procedure that sorts through potential detections following inference [30]. For identification, segmentation, and classification, there are five models (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x, respectively). Whereas YOLOv8x is the most precise yet the least fast of them all, YOLOv8 Nano is the fastest and smallest [31]. Figure 1 shows YOLO v8 architecture made by GitHub user RangeKing [32]. Differences from YOLOv5 are as follows [32]:

- C2f module used in place of C3 module.
- Change the Backbone's initial 6 × 6 Conv to a 3 × 3 Conv.
- Remove Convs Nos. 10 and 14 from the YOLOv5 configuration.

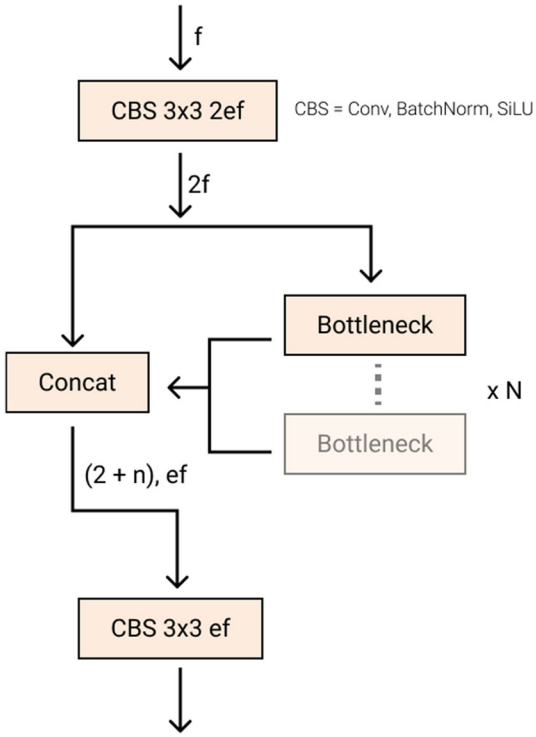


Fig. 1 YOLOv8 C2f module [30]

- Change the initial 1 × 1 Conv in the bottleneck to a 3 × 3 Conv.
- Remove the objectness step using the decoupled head.

The basic building block was altered, C2f replacing C3, and the initial 6 × 6 conv of the stem is replaced with a 3 × 3. Figure 2 is a diagram summarizing the module [30], where "f" represents the total amount of features, "*e*" is the rate of growth, and CBS is a block made up of a Conv, a BatchNorm, and a SiLU later. The initial conv's kernel dimension was adjusted from 1 × 1 to 3 × 3; nevertheless, the bottleneck is still identical as in YOLOv5.

3 Proposed framework

The primary goal of a smart city project is to enhance the intelligence of city systems and applications. This is achieved by incorporating various specifications and qualities, such as: (i) a secure and open access infrastructure that is both robust and scalable; (ii) an architecture strategy that is user- or citizen-centered; (iii) the ability to store, retrieve, share, tag, transport, and wear a vast amount of public and private data, thereby enabling people to access information as and when they require it; (iv) an analytical and integrative application level capability; and (v) intelligent physical and network infrastructure that facilitates complex and remote services and applications

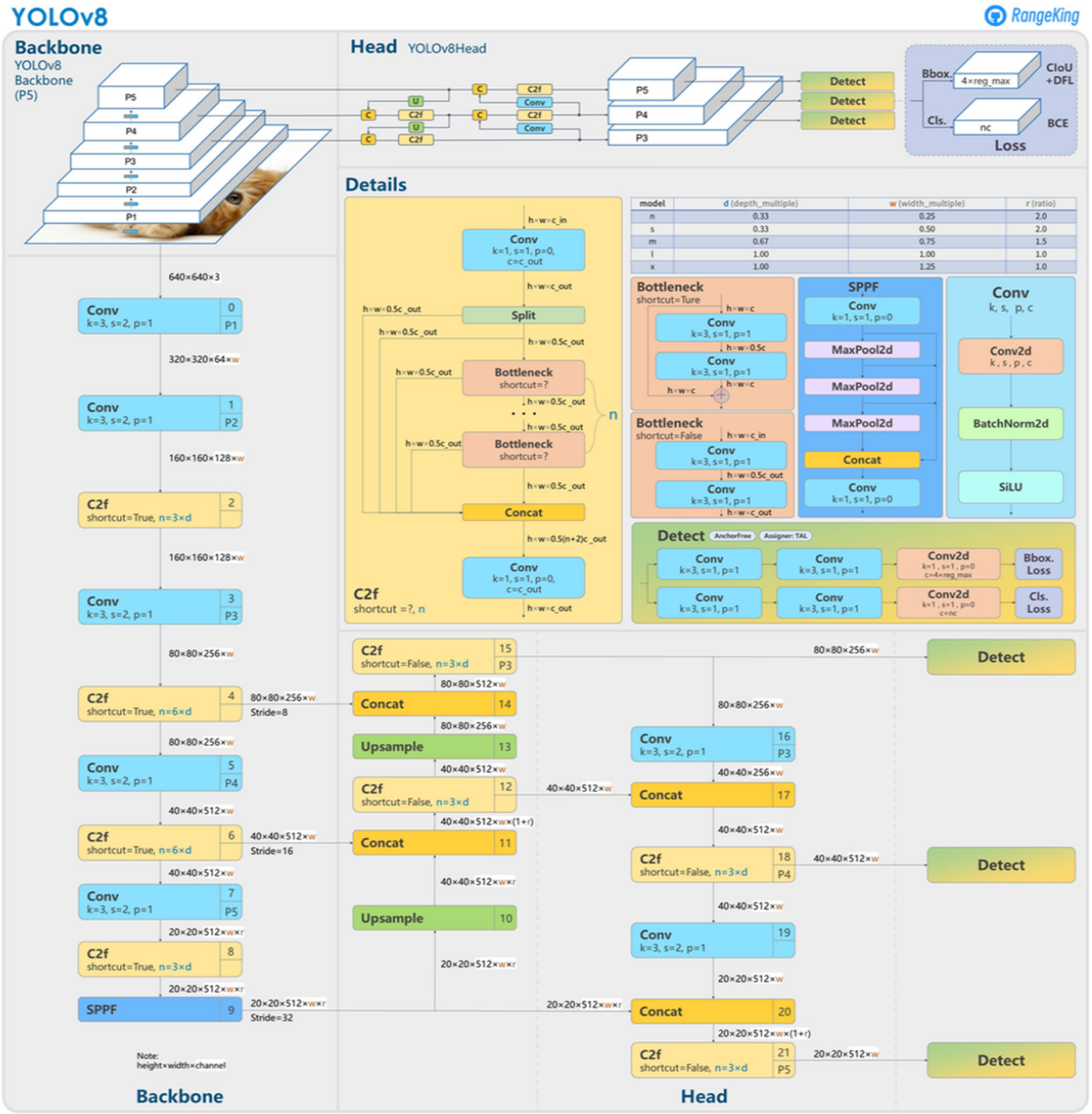


Fig. 2 YOLO v8 architecture [32]

while transferring large volumes of heterogeneous data. A framework is necessary to represent and structure the fundamental components, relationships, and data flow of multidisciplinary smart city projects. As depicted in Fig. 3, the deployment architecture for the proposed smart city project has a layered structure consisting of four tiers.

The proposed framework for a smart city consists of four primary layers: (i) Application layer, (ii) Fog layer, (iii) Cloud layer, and (iv) IoT layer.

3.1 Application layer

The Application layer is a crucial part of the proposed framework for a smart city, which focuses on specific applications such as the Smart Fire Detection System (SFDS). The SFDS can detect fire hazards in different locations, including government buildings, public areas, hospitals, residential areas, and highways, and ensure the safety of people and property. The Application layer plays a significant role in enabling the SFDS to serve various

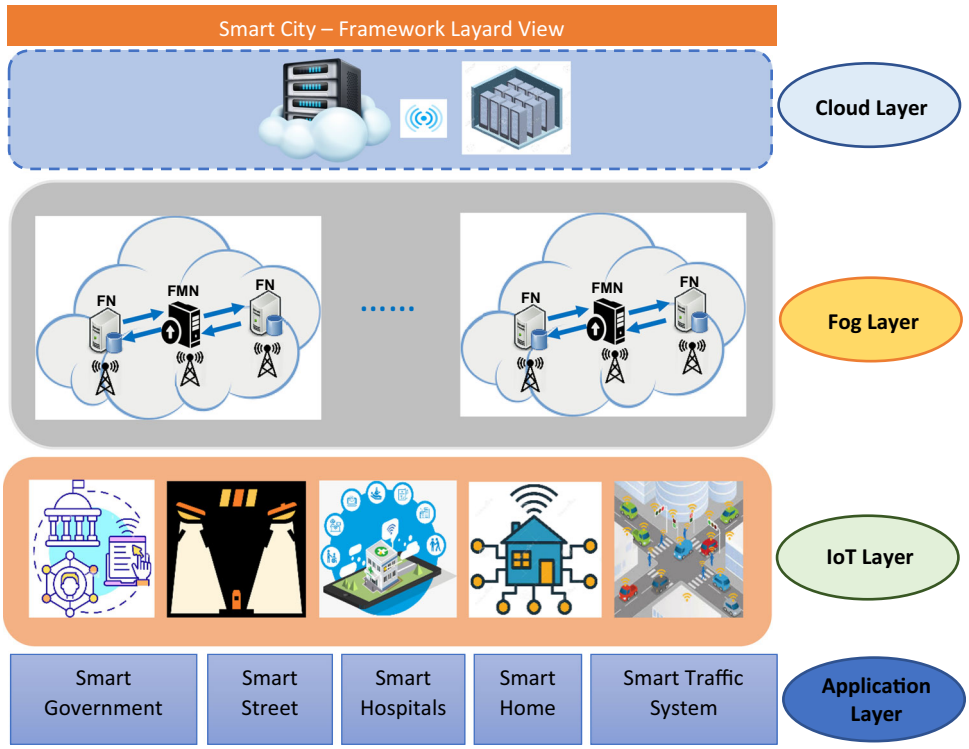


Fig. 3 The proposed smart city framework

applications, including Smart Government, Smart Street, Smart Hospitals, Smart Home, and Smart Traffic System.

3.2 Fog layer

The Fog layer is a bridge between the IoT layer and the Cloud layer in a smart city framework, responsible for processing data at the edge of the network in real-time, reducing network congestion and latency. It provides a more secure environment for data processing and offers features such as data management, analytics, and security. The Fog Master Node (FMN) is the main controller that receives data from the IoT layer and decides whether to process it in the Fog layer or transmit it to the Cloud.

3.3 Cloud layer

The Cloud layer is responsible for storing and processing big data generated by IoT devices in smart cities. It offers several benefits, including high scalability, cost-effectiveness, and reliability. In the proposed framework, if the desired data are not found in the fog cache, the request will be forwarded to the cloud for further processing. Cloud computing is also used to store data generated by the SFDS for future analysis and decision-making. Overall, the Cloud layer plays a critical role in enabling the SFDS to process and analyze large volumes of data efficiently.

3.4 IoT layer

The IoT layer in a smart city consists of physical devices, sensors, and actuators connected to the Internet that collect and exchange data. Sensors detect environmental changes, actuators control and move systems, and effectors interact with the environment. Data collected from sensors are stored and processed to analyze patterns, detect anomalies, and predict future events. Cloud and Fog computing is used to process the data in real-time, enabling the development of intelligent systems.

3.4.1 Smart fire detection system (SFDS) methodology

The SFDS methodology uses the YOLOv8 detection model, which offers fast and precise object detection without the need for a regional proposal network. The system is optimized to reduce the number of parameters needed for detection, making it more efficient. The SFDS uses computer vision to automatically detect fires in images and video streams. The methodology involves several steps, as shown in Fig. 4.

Step 1: Data collection This step involves gathering a large dataset of images and videos containing fire and non-fire images. This can be done by collecting images and videos from public sources, such as social media and news websites, or by capturing footage using specialized cameras or sensors. The dataset should be carefully curated, checked for duplicates, and labeled appropriately with the

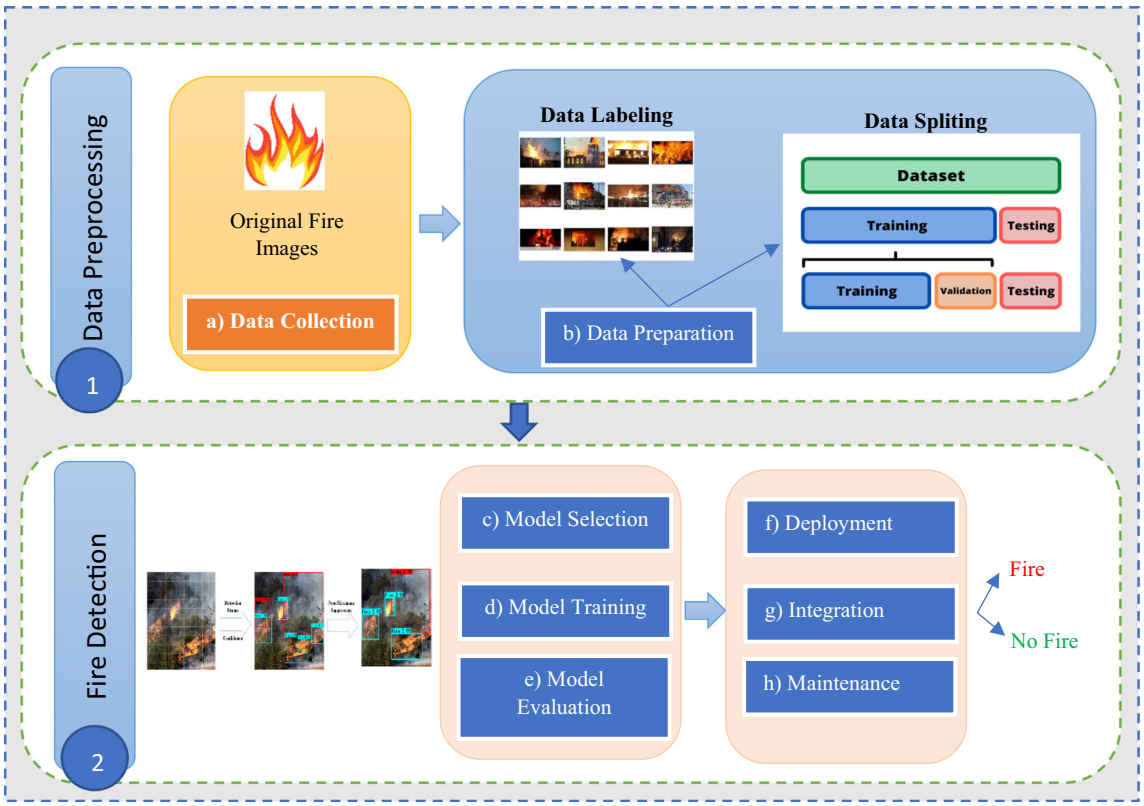


Fig. 4 Smart fire detection (SFD) system methodology

presence or absence of fires using automated labeling tools or manual labeling. It is important to ensure the dataset is balanced with an equal number of fire and non-fire images to prevent the model from becoming biased toward one class of images.

Step 2: Data preparation In this step, the collected dataset of fire and non-fire images and videos is prepared for training and testing the smart fire detection system. This involves labeling the images and videos with bounding boxes around the fires, which can be done manually or with a tool like LabelImg. The labeled data are then split into training and testing sets, ensuring both sets are representative of the overall dataset. Other pre-processing steps, such as resizing or normalizing the data, may also be necessary. The goal is to have a large enough, balanced dataset that can generalize well to new data.

Step 3: Model selection This step involves selecting the appropriate object detection algorithm for training the fire detection model. There are several algorithms to choose from, such as YOLOv8, Faster R-CNN, and SSD, each with its own advantages and disadvantages. The selected algorithm should have good performance on the collected dataset and be capable of handling different fire scenarios, depending on the requirements of the smart fire detection system. YOLOv8 is a popular choice due to its speed and accuracy, but other algorithms can also be used based on specific needs.

Step 4: Model training In this step, the YOLOv8 model is trained on the labeled dataset prepared in step 2. Model training involves teaching the deep learning model to recognize the features of fire and non-fire images and differentiate between them accurately. The YOLOv8 model is trained using a deep learning framework like TensorFlow or PyTorch, which provides the necessary tools and libraries to build and train neural networks.

Step 5: Model evaluation The trained model's performance is evaluated using various metrics such as accuracy, precision, recall, and F1 score. These metrics provide a measure of how well the model is performing in terms of identifying fires and non-fires. If the model's performance is not satisfactory, it can be fine-tuned by adjusting the hyperparameters or adding more training data. It is important to find a balance between overfitting and underfitting to ensure that the model generalizes well to new data.

Step 6: Deployment This step involves deploying the trained model in a real-time system that can process live video streams from cameras. A computer or server with high computing power and a GPU is required to process the video streams in real-time. The system should be able to read video frames from a camera or video stream, process

Table 2 Dataset descriptive information

Parameters	No. of images
Training	
Outdoor fire large and small	4375
Normal scene	3547
Smoky	6212
Fire with noise	2206
Normal scene with noise	1747
Smoke with noise	3129
Testing	
Outdoor large and small	1079
Normal scene	843
Smoky	1624
Fire with noise	521
Normal scene with noise	448
Smoke with noise	789

them through the trained model, and generate alerts when a fire is detected. False positives can be handled by using a threshold value, which determines the minimum confidence level required for the model to detect a fire, and detections below this threshold value are discarded as false positives.

Step 7: Integration This step involves integrating the fire detection system with other systems, such as fire alarms, sprinkler systems, and emergency response systems. When a fire is detected, the fire alarm system should be triggered to alert people in the building and evacuate them. The sprinkler system can also be activated to suppress the fire. In addition, emergency response systems can be notified with critical information such as the location and severity of the fire to provide timely and effective response. Proper integration of these systems is important to avoid false alarms and ensure a seamless and efficient response to fires. Testing and validation should be carried out to ensure the systems work together effectively.

Step 8: Maintenance This step involves maintaining the deployed fire detection system to ensure its effectiveness over time. This includes updating the model with new data, testing the system periodically, and maintaining the hardware and software components of the system. Regular maintenance helps to reduce the risk of false alarms and improve overall safety.

Fig. 5 Some images of the used dataset

3.4.2 Smart fire detection (SFD) algorithm

The Smart Fire Detection (SFD) Algorithm uses computer vision to detect fires in real time from live camera feeds or pre-recorded video files. As illustrated in Algorithm 1, it uses a pre-trained Yolov8 object detection model on a large dataset of fire and non-fire images. It takes a dataset of video frames as input and outputs detected objects, including fire-related classes such as “flames”, “smoke”, or “embers”. The algorithm loops through each frame of the video, applying pre-processing techniques to the current frame and passing it to the Yolov8 model for object detection. If a fire-related class is detected, the algorithm triggers an alarm and notifies the relevant authorities. Finally, the algorithm saves the output video with the detected objects highlighted. The SFD algorithm is a powerful tool for detecting fires in real-time and enables quick and effective responses to potential fire hazards.

The smart fire detection (SFD) uses Yolov8 object detection model. It involves several steps:

- The video input source is set up from either a live camera or pre-recorded video file.
- The video capture process is started, and each frame of the video is looped through.
- Image pre-processing techniques are applied to each frame, and the pre-processed frame is passed to the Yolov8 model for object detection.
- The detected objects are checked for fire-related classes, such as “flames”, “smoke”, or “embers”.
- If a fire-related class is detected, an alarm is triggered, and relevant authorities are notified.
- The video capture process is stopped, and the output video with the detected objects highlighted is saved.

Algorithm 1: Smart Fire Detection (SFD) Algorithm

- **Input: Dataset**
- **Output: Detected Objects**
- **Steps:**

1. Load the pre-trained Yolov8 object detection model and set up the necessary configurations.

```
# Load Yolov8 model and set up configurations
net = darknet.load_net_custom("cfg/yolov8.cfg",
"weights/yolov8.weights", 0, 1)
```

2. Set up the video input source, either from a live camera or from a pre-recorded video file.

```
# Set up video input source
cap = cv2.VideoCapture(0)
```

3. Start the video capture process.
4. Loop through each frame of the video:
   a. Apply image pre-processing techniques, such as 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, such as "flames," "smoke," or "embers."
   d. If a fire-related class is detected, trigger an alarm and notify the relevant authorities.

```
# Start video capture process
while True:
    # Read current frame from video input source
    ret, frame = cap.read()

    # Apply image pre-processing techniques
    resized_frame = cv2.resize(frame, (416, 416))
```

5. Stop the video capture process.
6. Save the output video with the detected objects highlighted.

Table 3 YOLO v8 configuration parameters

Parameters	Values
Epoch	300
Learning rate	0.01
Image size	512
Batch size	16
Number of images	26,520
Layers	225
Parameters	11,136,374

4 Implementation and evaluation

This section discusses the used dataset, the performance metrics, and the performance evaluation.

4.1 Used dataset

The dataset contains a diverse range of fire and smoke scenarios, including indoor and outdoor fires, small and large fires, low-light and high-light conditions, and normal scenes without fire. The data-set contains 26,520 images including images that have fires, smoke, fire and smoke, and normal scenes without fire or smoke. The dataset is divided into 21,216 for training and 5304 for testing. Despite the large dataset size, the proposed method proved its superiority for smoke and fire detection with high accuracy. Besides, it can generalize and validate new images with very high accuracy. Table 2 describes dataset description of number of images for large and small fires with and without noise, smoky with and without noise, and normal scenes. Figure 5 shows samples from the used dataset.

A YOLO v8 model has been trained on the dataset using transfer learning, where we initialized the model with pre-trained weights on the COCO dataset and fine-tuned it on our dataset. We used a batch size of 16 and trained the model for 300 epochs with an initial learning rate of 0.01. Table3 describes the configuration parameters of YOLO v8.

The proposed model has been implemented, trained, and validated on Kaggle using GPU platform. From the experimental results, it is shown that our model can detect fires with precision 95.7% and smoke with precision 99.3% with overall mean average precision (mAP) for the two classes 97.5%. Figure 6 shows overall results of the proposed model including loss, precision, and recall. Figure 7A, B shows precision recall curve. Also, Fig. 8 presents the results of fire and smoke detection with the proposed model.

4.2 Performance metrics

The weighted average between recall percentages and precision percentages means is known as the FM score. Consequently, this score takes both false positives and false negatives into account. Although FM is more prevalent than precision, accuracy is not immediately simple to understand. When false positives and false negatives have comparable costs, accuracy performs well. Consideration of recall as well as accuracy is preferable if the costs of false positives and false negatives vary. In terms of positive findings, precision is the proportion of accurately predicted observations to all predicted positive findings. The recall is

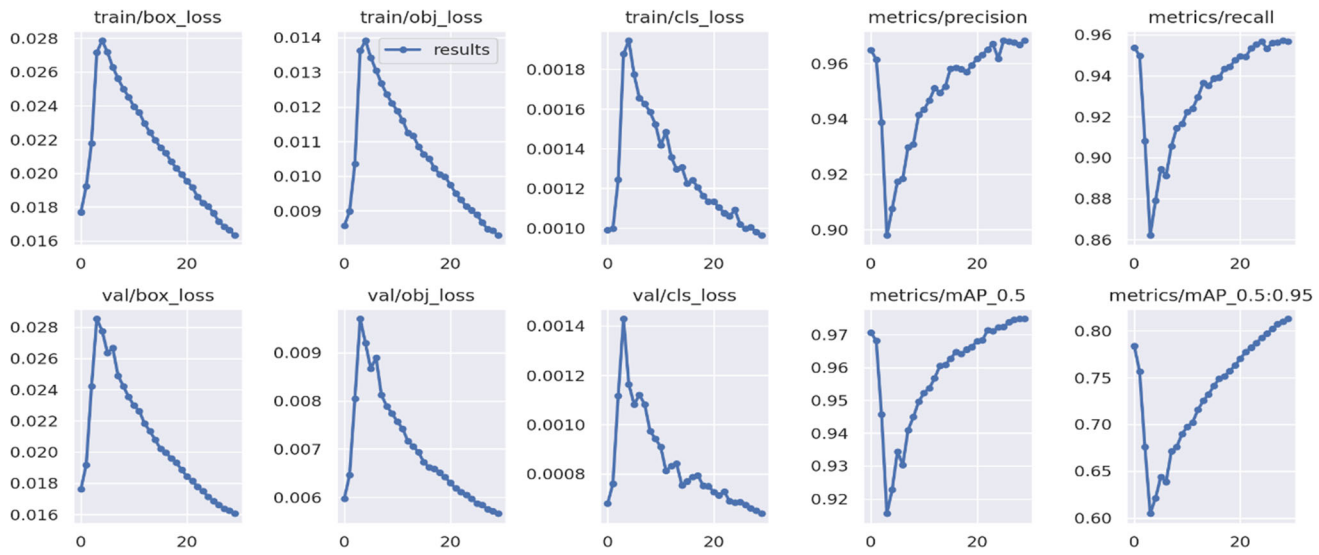
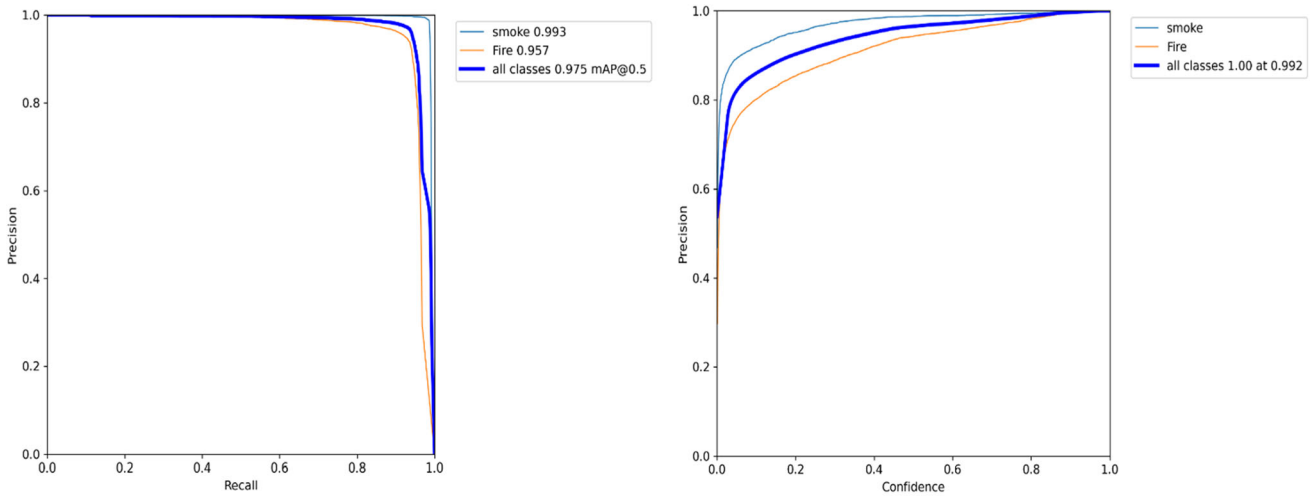
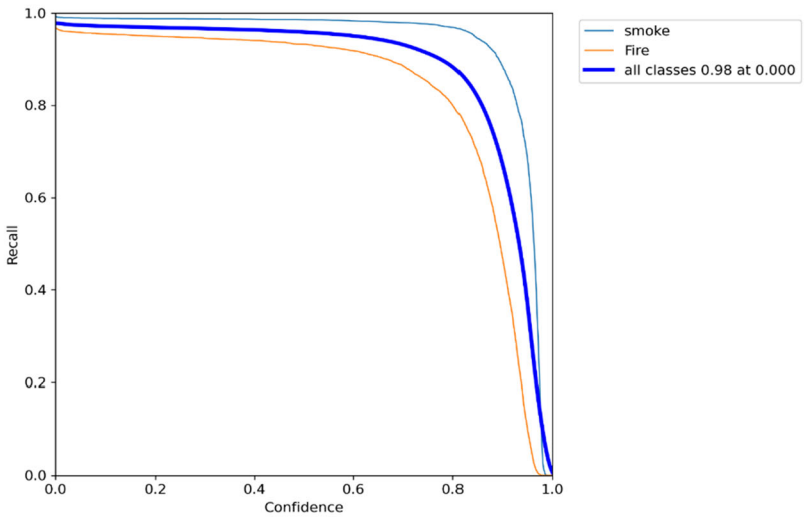


Fig. 6 The results of the proposed model



(A) Precision- Recall curve of the proposed model



(B) Precision- Recall curve of the proposed model

Fig. 7 Precision–recall curve of the proposed model

the proportion of true positive predictions over all actual positives. It can be calculated as in Eq. (1). The precision is the proportion of true positive predictions over all positive predictions. It can be calculated as in Eq. (2)

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

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

where TP stands for True Positive. FP stands for False Positive, TN stands for True Negative, and FN stands for False Negative. Precision and recall are considered while calculating F-Measure, abbreviated as FM as calculated in Eq. (3):

$$\text{FM} = 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}} \tag{3}$$

4.3 Results and discussion

The proposed system is evaluated using the standard metrics of precision, recall, and F1 score. Besides, dataset size is considered in comparison and previous methods consider fire, smoke, or both. We also compared our system with two state-of-the-art fire detection systems: the fire detection system based on deep learning [2], A YOLOv6-Based Improved Fire Detection Approach [33], Real-time video fire/smoke detection-based YOLO v2 [34], A smoke detection model based on improved YOLOv5 [35], and Fire detection method in smart city environments using a deep learning-based approach.

Table 4 and Fig. 9 show the performance comparison of the three systems on the test dataset. The proposed system outperformed existing systems with precision, recall, and F1 score, demonstrating its effectiveness in detecting fires

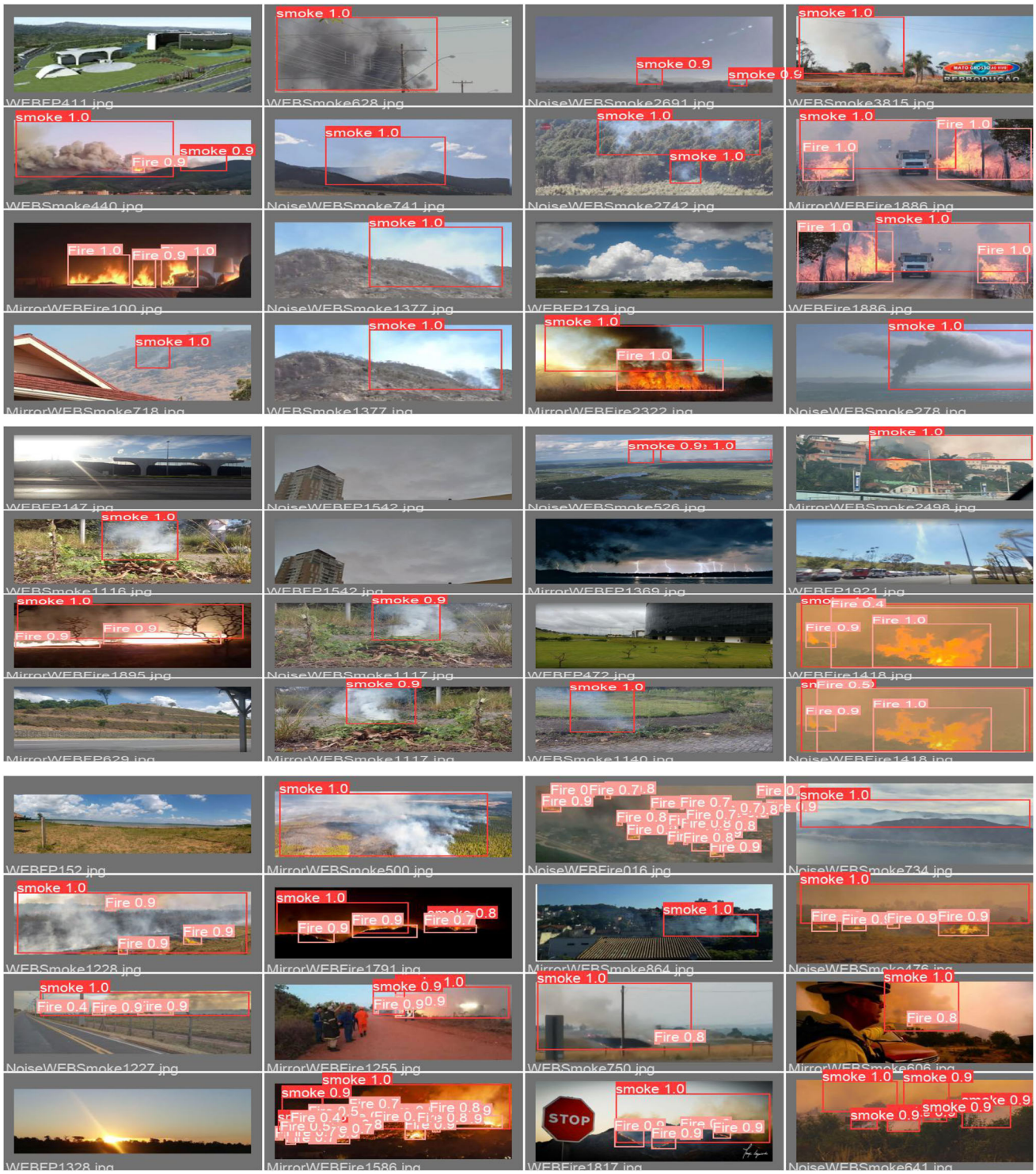


Fig. 8 The result of proposed model for fire and smoke detection

in real-world scenarios. Regarding the proposed methods (they have the same dataset) in [4, 35], they achieve higher performance metrics, but it is implemented in a smaller dataset size of 9200 compared to our dataset with 26,520 images. In addition, they are proposed for detecting fire only without considering smoke. However, our proposed model can detect fire and smoke with a high precision rate of 97.1% for all classes with a smoke precision of 99.3% and a fire precision of 95.7%. Moreover, it can generalize new data very accurately as it is implemented in a large

Table 4 Performance comparison of fire detection systems

System	Model	Precision %	Recall %	*F*1 score	Dataset size	Fire/smoke
Norkobil Saydirasulovich [33]	YOLO v6	93.48	28.29	0.9	4000	Fire/smoke
Saponara [34]	YOLO v2	97	97	95.4	400	Fire/smoke
Saponara [35]	YOLO v5	94.99	78.28	0.858	20,000	smoke
Abdusalomov [36]	YOLO v3	98.1	99.2	0.995	9200	Fire
Kuldoshbay [4]	YOLO v4	98.2	99.7	0.997	9200	Fire
Proposed Model	YOLO v8	97.5	95.7	0.962	26,520	Fire/smoke
		99.3 smoke and 95.7 fire				

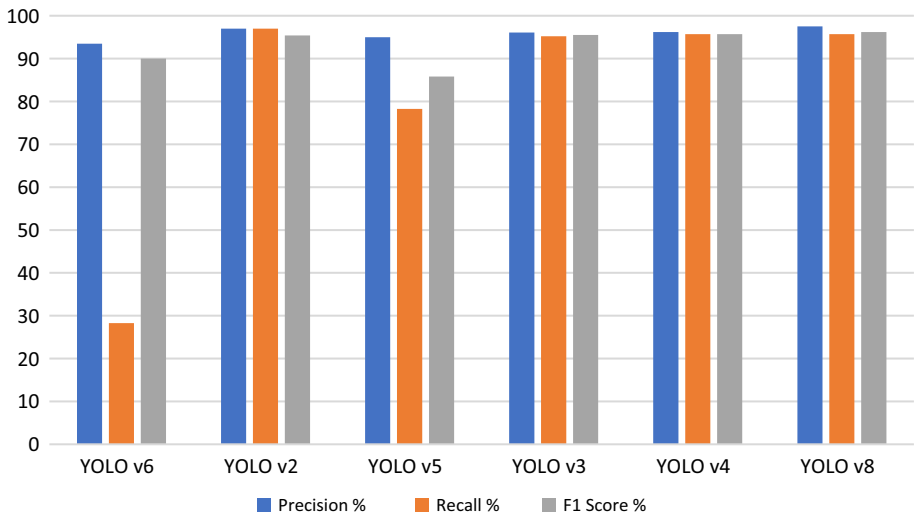


Fig. 9 The performance comparison of fire detection systems

dataset containing a diverse range of fire and smoke scenarios, including indoor and outdoor fires, small and large fires, low-light and high-light conditions, and normal scenes without fire. We attribute the superior performance of our system to the improved accuracy and speed of YOLO v8 compared to the earlier versions of YOLO and other deep learning-based fire detection systems.

5 Conclusion

This paper proposed an improved fire detection approach for smart cities based on the YOLOv8 algorithm, called the smart fire detection system (SFDS), which leverages the strengths of deep learning to detect fire-specific features in real time. The SFDS approach has the potential to improve the accuracy of fire detection, reduce false alarms, and be cost-effective compared to traditional fire detection methods. It can also be extended to detect other objects of interest in smart cities, such as gas leaks or flooding. The proposed system utilizes a deep neural network that is trained on a large dataset of images containing fires to detect and locate fires in real time with a high precision rate of 97.1% for all classes. The evaluation of the system on a benchmark dataset showed that it achieves high accuracy and outperforms existing fire detection methods. The proposed system is robust to various environmental conditions, such as smoke, and can detect fires at different scales and orientations. The proposed fire detection system has potential applications in various fields, such as public safety, industrial safety, and environmental monitoring. It can aid in the early detection and mitigation of fires, potentially saving lives and minimizing damage to property and the environment. Overall, this paper provides a valuable contribution to the field of computer vision and real-time fire detection, and the proposed system can be further improved and applied in real-world scenarios. In the future work, we can use OCNN [37] to achieve better results as it achieved a good performance in [38–44]. We can also use correlation methods like [45].

Author contributions It is a collaborative effort where FMT and HZ worked together. HZ came up with the idea and wrote the abstract and

the proposal, while FMT contributed to making comparisons and made the experiments.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). The authors received no specific funding for this study.

Data availability Data will be available on request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval There is no any ethical conflict.

References

1. Gohari A et al (2022) Involvement of surveillance drones in smart cities: a systematic review. IEEE Access 10:56611–56628
2. Huang PY, Chen YT, Wu CC (2019) A fire detection system for smart buildings based on deep learning. J Intell Fuzzy Syst 37(4):5541–5551
3. Chung T, Ball S, Stentz A (2018) Early fire detection using machine learning in smart buildings. In: Proceedings of the 21st international conference on information fusion (FUSION), pp 1–8
4. Avazov K et al (2021) Fire detection method in smart city environments using a deep-learning-based approach. Electronics 11(1):73
5. Zhang F, Zhao P, Xu S, Wu Y, Yang X, Zhang Y (2020) Integrating multiple factors to optimize watchtower deployment for wildfire detection. Sci Total Environ 737:139561
6. Karthi M et al (2023) Forest fire detection: a comparative analysis of deep learning algorithms. In: 2023 International conference on artificial intelligence and knowledge discovery in concurrent engineering (ICECONF). IEEE
7. El-Hosseini M et al (2021) A fire detection model based on power-aware scheduling for IoT-sensors in smart cities with partial coverage. J Ambient Intell Humaniz Comput 12(2021):2629–2648
8. Khan S, Khan A (2022) FFireNet: deep learning based forest fire classification and detection in smart cities. Symmetry 14(10):2155
9. Mukhiddinov M, Abdusalomov AB, Cho J (2022) A wildfire smoke detection system using unmanned aerial vehicle images based on the optimized YOLOv5. Sensors 22(23):9384
10. Avazov K, Mukhiddinov M, Makhmudov F, Cho YI (2022) Fire detection method in smart city environments using a deep-learning-based approach. Electronics 11:73. https://doi.org/10.3390/electronics11010073
11. Al-Turjman F, Al-Karaki JN, Al-Bzoor Z (2021) Hybrid deep learning-based approach for fire detection in smart cities. Sensors 21(6):2186
12. Huang J, Luo Q, Wang J, Guo J (2020) Fire detection in outdoor scenes using YOLOv3. IEEE Access 8:114978–114985
13. Jia J, Cao Y, Wang X, Huang J (2019) Fire detection based on multi-model fusion. In: 2019 4th International conference on image, vision and computing (ICIVC), pp 329–332. IEEE
14. Wang F, Li J, Li Y, Li Y, Huang Y (2018) Real-time fire detection in surveillance video using YOLOv2. In: 2018 13th IEEE conference on industrial electronics and applications (ICIEA), pp 2428–2432. IEEE
15. He et al. xxx
16. Shen Y, Liu J, Zhao G, Li X (2017) A deep learning approach for fire detection using convolutional neural networks. IEEE Access 5:13251–13258
17. Ba R, Chen C, Yuan J, Song W, Lo S (2019) SmokeNet: satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention. Remote Sens 11:1702
18. Luo Y, Zhao L, Liu P, Huang D (2018) Fire smoke detection algorithm based on motion characteristic and convolutional neural networks. Multimed Tools Appl 77:15075–15092
19. Sharma J, Granmo OC, Goodwin M (2021) Emergency analysis: multitask learning with deep convolutional neural networks for fire emergency scene parsing. In: Fujita H, Selamat A, Lin JCW, Ali M (eds) Advances and trends in artificial intelligence. Artificial Intelligence Practices; IEA/AIE 2021. Lecture Notes in Computer Science. Springer, Cham, vol 12798
20. 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
21. Peruzzi G, Pozzebon A, Van Der Meer M (2023) Fight fire with fire: detecting forest fires with embedded machine learning models dealing with audio and images on low power IoT devices. Sensors 23(2):783
22. Khan A et al (2022) CNN-based smoker classification and detection in smart city application. Sensors 22(3):892
23. Biswas A, Ghosh SK, Ghosh A (2023) Early fire detection and alert system using modified inception-v3 under deep learning framework. Procedia Comput Sci 218:2243–2252
24. Liu G, Yuan H, Huang L (2023) A fire alarm judgment method using multiple smoke alarms based on Bayesian estimation. Fire Saf J 136:103733
25. Li P, Zhao W (2020) Image fire detection algorithms based on convolutional neural networks. Case Stud Therm Eng 19:100625
26. 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
27. Pan H, Badawi D, Cetin AE (2020) Computationally efficient wildfire detection method using a deep convolutional network pruned via Fourier analysis. Sensors 20:2891
28. Li T, Zhao E, Zhang J, Hu C (2019) Detection of wildfire smoke images based on a densely dilated convolutional network. Electronics 8:1131
29. Kim B, Lee J (2019) A video-based fire detection using deep learning models. Appl Sci 9:2862
30. https://blog.roboflow.com/whats-new-in-yolov8/#what-is-yolov8
31. https://learnopencv.com/ultralytics-yolov8/#YOLOv8-vs-YOLOv5
32. https://github.com/ultralytics/ultralytics/issues/189
33. Norkobil Saydirasulovich S et al (2023) A YOLOv6-based improved fire detection approach for smart city environments. Sensors 23(6):3161

34. Saponara S, Elhanashi A, Gagliardi A (2021) Real-time video fire/smoke detection based on CNN in antifire surveillance systems. J Real-Time Image Proc 18:889–900
35. Wang Z et al (2022) A smoke detection model based on improved YOLOv5. Mathematics 10(7):1190
36. Abdusalomov A et al (2021) An improvement of the fire detection and classification method using YOLOv3 for surveillance systems. Sensors 21(19):6519
37. Talaat FM, Gamel SA (2022) RL based hyper-parameters optimization algorithm (ROA) for convolutional neural network. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-022-03788-y
38. Talaat FM, Ali SH, Saleh AI, Ali HA (2020) Effective cache replacement strategy (ECRS) for real-time fog computing environment. Clust Comput. https://doi.org/10.1007/s10586-020-03089-z
39. Hassan E, El-Rashidy N, Talaat FM (2022) Review: mask R-CNN models. https://doi.org/10.21608/njccs.2022.280047
40. ZainEldin H, Gamel SA, El-Kenawy ES, Alharbi AH, Khafaga DS, Ibrahim A, Talaat FM (2022) Brain tumor detection and classification using deep learning and sine-cosine fitness grey wolf optimization. Bioengineering 10(1):18. https://doi.org/10.3390/bioengineering10010018
41. El-Rashidy N, Ebrahim N, El Ghamry A, Talaat FM (2022) Prediction of gestational diabetes based on explainable deep learning and fog computing. Soft Comput. https://doi.org/10.1007/s00500-022-07420-1
42. El-Rashidy N, Ebrahim N, El Ghamry A, Talaat FM (2022) Utilizing fog computing and explainable deep learning techniques for gestational diabetes prediction. Neural Comput Appl. https://doi.org/10.1007/s00521-022-08007-5
43. El-Balka RM et al (2022) Enhancing the performance of smart electrical grids using data mining and fuzzy inference engine. Multimed Tools Appl 81(23):33017–33049
44. Talaat FM (2022) Effective deep Q-networks (EDQN) strategy for resource allocation based on optimized reinforcement learning algorithm. Multimed Tools Appl 81:39945–39961
45. Alshathri S, Talaat FM, Nasr AA (2022) A new reliable system for managing virtual cloud network. Comput Mater Continua 73(3):5863–5885. https://doi.org/10.32604/cmc.2022.026547