

# Enhancing fire detection capabilities: Leveraging you only look once for swift and accurate prediction

Agung Nugroho<sup>1</sup>, I Made Artha Agastya<sup>2</sup>, Kusrini<sup>1</sup>

<sup>1</sup>Magister of Informatics, Universitas AMIKOM Yogyakarta, Yogyakarta, Indonesia

<sup>2</sup>Informatics, Faculty of Computer Science, Universitas AMIKOM Yogyakarta, Yogyakarta, Indonesia

## Article Info

### Article history:

Received Dec 27, 2023

Revised Nov 4, 2024

Accepted Nov 14, 2024

### Keywords:

CSP-DarkNet53

Deep neural network

Feature extractor

Fire detection

YOLO

## ABSTRACT

Detecting fires is crucial to prevent potentially catastrophic outcomes. Traditional fire detection methods, relying on electronic, chemical, or mechanical sensors, often suffer from time delays in activation due to threshold parameters. An emerging alternative utilizes artificial intelligence, particularly image-based fire detection, using convolutional neural networks (CNNs). You only look once (YOLO) is a state-of-the-art object detection framework prized for speed and real-time capabilities. In our research, we conducted multiple training experiments employing various deep neural network (DNN) architectures as feature extractors for object detection within the YOLOv5 framework. These architectures included MobileNetV3, ResNet, and CSP-Darknet53. Among these configurations, YOLOv5 with CSP-Darknet53 (scale s) emerged as the most accurate, boasting mAP@50 of 0.88 and an impressive FPS of 73, with training model size of 14.50 MB. Furthermore, we integrated the selected model with the streamlit package to create a user-friendly web application interface for fire detection testing. The resulting model demonstrates remarkable efficiency, detecting fires within 0.01 seconds. This research represents a significant advancement in fire detection technology, offering both rapid detection and enhanced accuracy, with potential applications in various settings, from indoor facilities to outdoor environments.

This is an open access article under the [CC BY-SA](#) license.



## Corresponding Author:

I Made Artha Agastya

Informatics Study Program, Faculty of Computer Science, Universitas AMIKOM Yogyakarta

St. Ringroad Utara, Condong Catur, Depok, Sleman, Yogyakarta 55283, Indonesia

Email: artha.agastya@amikom.ac.id

## 1. INTRODUCTION

Fire is a disaster that often occurs due to human negligence or force majeure [1]–[3]. Fire causes significant property damage and poses severe risks to human lives and the environment. Prevention measures such as education, proper infrastructure, and effective firefighting strategies are crucial in mitigating its impact. Fire detection methods are generally carried out using conventional methods based on electronic sensors and chemicals, and further methods based on artificial intelligence based on image processing [4]–[6]. The electronic sensor method delays providing a warning [7]–[9] because the parameters for activating the alarm must be at sufficient levels so that the sensor can work. When the sensor reads environmental parameters that trigger the warning, the fire object has grown and is more difficult to control. Fires that cause harm to both the environment and humans [10] occur because the flames cannot be detected as quickly as possible.

Pushpa and Kamarasan [11] used a merged Gaussian mixture model (MGMM) for the process of extracting key frames and kernel support vector machine (KSVM) to determine possible areas of fire objects.

The result was an average detection rate of 90.56%. Mahmoud *et al.* [12] created a fire detection model based on transfer learning using a convolutional neural network (CNN) architecture, producing fire detection with faster and more efficient training time. In the Bari *et al.* investigation [13], deep learning-based detection was made with surveillance cameras using the InceptionV3 and MobileNetV2 architecture using the transfer learning method. It was concluded that the fire detection training process using the transfer learning method could be carried out more quickly and using a smaller dataset. Cheng [14] uses the recurrent convolutional neural network (R-CNN) architecture to detect fire and smoke, and it is concluded that R-CNN-based fire and smoke detection can increase detection accuracy and reduce false alarms. Muhammad *et al.* [7] conducted fire detection research based on CNN architecture to detect fire on surveillance cameras. Their research showed that the resulting model could outperform manual-based fire detection with sensors and fire detection based on the AlexNet architecture. Zheng *et al.* [15] implemented YOLOv3 and produced a detection speed of 27 FPS, indicating real-time detection.

The evolution of you only look once (YOLO) models represents significant progress in object detection. Introduced in 2016 by Joseph Redmon, YOLOv1 [16] revolutionized detection by framing it as a single regression problem, enabling real-time performance, though it struggled with smaller objects and overlapping instances. YOLOv2 [17], also called YOLO9000, improved upon this with anchor boxes, batch normalization, and simultaneous classification and detection training. YOLOv3 [18], released in 2018, incorporated multi-scale predictions with a deeper architecture (Darknet-53) for better handling of small objects and multi-label classification. In 2020, Alexey Bochkovskiy led the development of YOLOv4 [19], balancing speed and accuracy by introducing CSPDarknet53, Mosaic augmentation, and CIoU loss. That same year, Ultralytics released YOLOv5 [20], departing from traditional academic channels to focus on practicality with a PyTorch-based framework, offering model size variants, auto-anchor learning, and deployment support for edge devices.

Research by Mahmoud *et al.* [12] can detect fire objects from image data and is more inclined towards classifying images with fire labels and non-fire labels with deep neural network (DNN) architecture with AlexNet architecture weights. The effectiveness of training is mentioned in the research by transferring AlexNet weights into the DNN process to increase classification accuracy from 90.3% to 96.7%. Bari *et al.* [13] conducted experiments with the InceptionV3 and MobileNetV2 architectures, with the first scheme being full training without freezing layers. The research results on the MobileNetV2 architecture with transfer learning with two unfrozen layers are F1 of 0.88, while the MobileNetV2 Full training scheme gets an F1 score of 0.83. For the InceptionV3 architecture, the entire training scheme gets an F1 Score of 0.75. Research conducted by Cheng [14] using the Fast R-CNN architecture obtained an accuracy of 96.78% and was better when compared to the AlexNet, ZF-Net, MCLBP, and HLTPMC architectures. Muhammad *et al.* [7] used the GoogleNet architecture, which was fine-tuned for training on a fire image dataset. The model's results produced an accuracy of 94.43%, outperforming other compared schemes.

Our research introduces a methodology centered on the strategic selection of the YOLOv5 backbone, marking a pivotal shift in fire detection capabilities for indoor and outdoor environments. Our investigation entailed a thorough exploration, evaluating 12 distinct backbone architectures to identify the most promising candidates. This comprehensive analysis highlights our unwavering dedication to pushing the boundaries of fire detection technology. The outcome of our endeavors not only promises quicker detection but also a significant enhancement in accuracy, signifying a remarkable advancement in the field. In the next section, the research methods used are explained, including the architectural configuration used, namely YOLOv5 with standard backbones or modifications with other DNNs such as ResNet-50, ResNet-101, ResNet152, ResNext-50, ResNetxt-101, and MobileNetV3-Large.

## 2. METHOD

The YOLOv5 [21] architecture consists of four blocks: the input block, backbone block, neck block, and head block (dense prediction), as shown in Figure 1. The input block is where the backbone will process the image. In YOLOv5, the default image size is 640 pixels. Backbone is a DNN architecture used for feature extraction. In backbone block, cross stage partial (CSP) approach is used in reducing spatial resolution and increasing its channel resolution is carried out. The neck block is the part for extracting feature pyramids to identify objects of several sizes and scales. The head is the final part of the YOLOv5 architecture and is responsible for implementing classes, confidence scores, and bounding boxes.

YOLOv5 was the YOLO version developed by Ultralytics and launched on January 10, 2023. Nadeem *et al.* [22] stated that the deeper the layer used in plain networks, the greater the error. For the results of architectural validation research using the COCO dataset, the VGG-16 architecture produces a mAP@50 of 41.5, while ResNet-101 produces a mAP@50 of 48.4. The ResNet architecture can be seen in Figure 2.

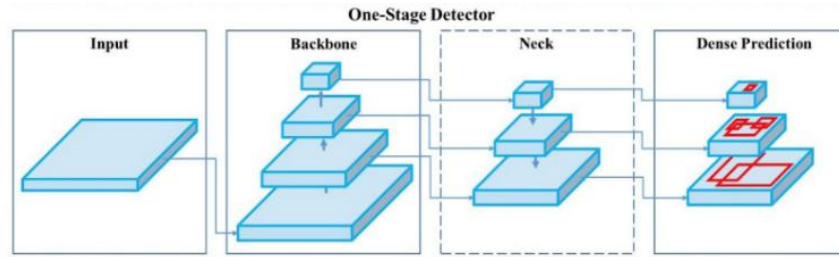


Figure 1. YOLO architecture blocks

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		$1.8 \times 10^9$	$3.6 \times 10^9$	$3.8 \times 10^9$	$7.6 \times 10^9$	$11.3 \times 10^9$

Figure 2. ResNet architectural blocks

Howard *et al.* [23] offers a solution in the form of lite reduced atrous spatial pyramid pooling (LR-ASPP) with results increasing accuracy by 3.2% compared to MobileNetV2 in ImageNet Classification, for the MobileNetV3-Large architecture as seen in Figure 3 and the MobileNetV3 block as seen in Figure 4. Our research used a fire object dataset to run on the YOLOv5 architectures obtained from the RoboFlow site [24]. The experiment used the Google Collab Pro platform, Runtime T4 GPU High RAM with Google Drive storage media. The research was carried out using several schemes, as depicted in the flow diagram in Figure 5. As explained in Figure 1, YOLO has a block backbone, block neck, and blockhead. In this study, the block backbone was modified with several DNNs, which are ResNet-50, ResNet-101, ResNet152, ResNext-50, ResNetxt-101, and MobileNetV3-Large.

Input	Operator	exp size	#out	SE	NL	s
$224^2 \times 3$	conv2d	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	-	RE	1
$112^2 \times 16$	bneck, 3x3	64	24	-	RE	2
$56^2 \times 24$	bneck, 3x3	72	24	-	RE	1
$56^2 \times 24$	bneck, 5x5	72	40	✓	RE	2
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 3x3	240	80	-	HS	2
$14^2 \times 80$	bneck, 3x3	200	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	480	112	✓	HS	1
$14^2 \times 112$	bneck, 3x3	672	112	✓	HS	1
$14^2 \times 112$	bneck, 5x5	672	160	✓	HS	2
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	conv2d, 1x1	-	960	-	HS	1
$7^2 \times 960$	pool, 7x7	-	-	-	-	1
$1^2 \times 960$	conv2d 1x1, NBN	-	1280	-	HS	1
$1^2 \times 1280$	conv2d 1x1, NBN	-	k	-	-	1

Figure 3. MobileNetV3-large architecture

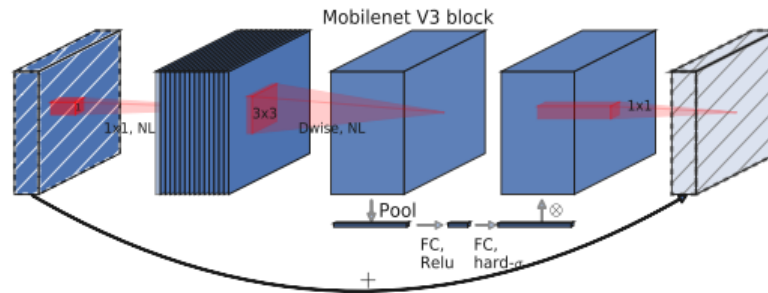


Figure 4. MobileNetV3 blocks

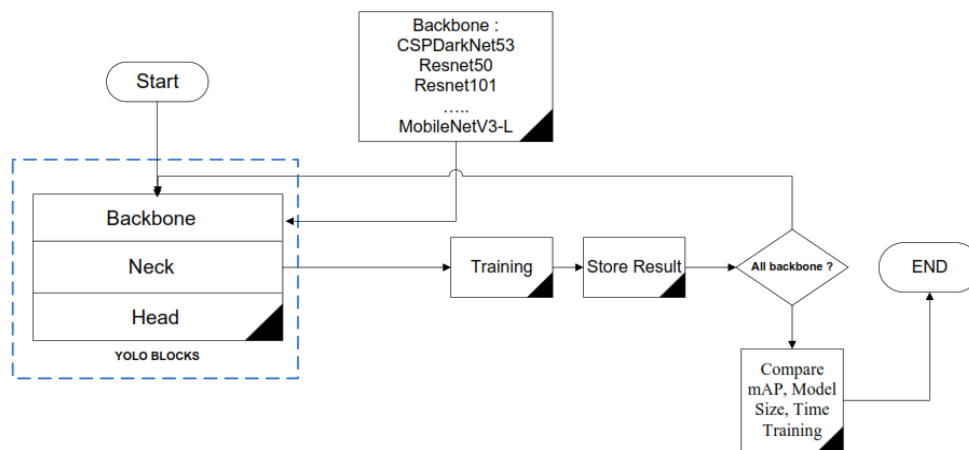


Figure 5. Research process flow

Our research modified the original YOLOv5 backbones with DNN architecture based on ResNet50, ResNet101, ResNet152, and MobileNetV3-Large. Each base DNN architecture with standard YOLO heads is trained. The results of all experiments are compared with the parameters, which are mAP@50, time train, and model size in MB. The original YOLOv5 backbone is illustrated in Figure 6. Figure 7 showcases an example of a ResNet backbone, while Figure 8 presents the configuration of a MobileNet backbone. The computer vision-based fire detection scheme is a model obtained from the training process integrated into the control room or security room computer. The YOLO models selected from the training results are used to monitor CCTV-captured videos of fire or smoke objects, as seen in Figure 9.

```
# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]], # 0-P1/2
  [-1, 1, Conv, [128, 3, 2]], # 1-P2/4
  [-1, 3, C3, [128]],
  [-1, 1, Conv, [256, 3, 2]], # 3-P3/8
  [-1, 6, C3, [256]],
  [-1, 1, Conv, [512, 3, 2]], # 5-P4/16
  [-1, 9, C3, [512]],
  [-1, 1, Conv, [1024, 3, 2]], # 7-P5/32
  [-1, 3, C3, [1024]],
  [-1, 1, SPPF, [1024, 5]], # 9
  ]
```

Figure 6. Standard YOLOv5 backbone configuration

```
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 7, 2, 3]], # 0
  [-1, 1, nn.BatchNorm2d, [None]], # 1
  [-1, 1, nn.ReLU, [True]], # 2
  [-1, 1, nn.MaxPool2d, [3, 2, 1]], # 3
  [-1, 3, resLayer, [64, 1, 1, 64, True]], # 4
  [-1, 4, resLayer, [128, 2, 1, 64, True]], # 5
  [-1, 6, resLayer, [256, 2, 1, 64, True]], # 6
  [-1, 3, resLayer, [512, 2, 1, 64, True]], # 7
  ]
```

Figure 7. Resnet50 backbone model.yaml configuration

```

[[-1, 1, conv_bn_hswish, [16, 2]], # 0-p1/2
[-1, 1, MobileNetV3_InvertedResidual, [ 16, 16, 3, 1, 0, 0]], # 1-p1/2
[-1, 1, MobileNetV3_InvertedResidual, [ 24, 64, 3, 2, 0, 0]], # 2-p2/4
[-1, 1, MobileNetV3_InvertedResidual, [ 24, 72, 3, 1, 0, 0]], # 3-p2/4
[-1, 1, MobileNetV3_InvertedResidual, [ 40, 72, 5, 2, 1, 0]], # 4-p3/8
[-1, 1, MobileNetV3_InvertedResidual, [ 40, 120, 5, 1, 1, 0]], # 5-p3/8
[-1, 1, MobileNetV3_InvertedResidual, [ 40, 120, 5, 1, 1, 0]], # 6-p3/8
[-1, 1, MobileNetV3_InvertedResidual, [ 80, 240, 3, 2, 0, 1]], # 7-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [ 80, 200, 3, 1, 0, 1]], # 8-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [ 80, 184, 3, 1, 0, 1]], # 9-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [ 80, 184, 3, 1, 0, 1]], # 10-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [112, 480, 3, 1, 1, 1]], # 11-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [112, 672, 3, 1, 1, 1]], # 12-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [160, 672, 5, 1, 1, 1]], # 13-p4/16
[-1, 1, MobileNetV3_InvertedResidual, [160, 672, 5, 2, 1, 1]], # 14-p5/32
[-1, 1, MobileNetV3_InvertedResidual, [160, 960, 5, 1, 1, 1]], # 15-p5/32
]

```

Figure 8. MobileNetV3-large backbone

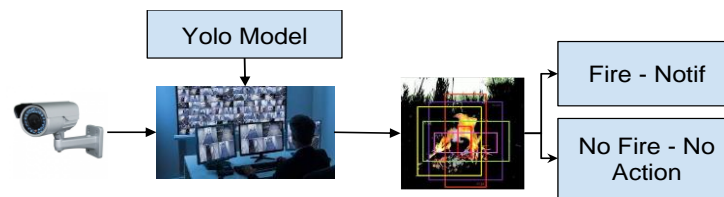


Figure 9. Computer vision-based fire detection scheme

### 3. RESULTS AND DISCUSSION

#### 3.1. Backbone comparison

It is more about calculating image data, whether it has a fire label or no fire. Object detection research based on YOLO was carried out in our study, not just image data classification. Several schemes are carried out to get the best object detection accuracy with the mean average precision (mAP) metric. As a result of research using several methods and configurations, the following comparison results were obtained in Table 1.

In real-world scenarios, detecting fires is crucial to alert authorized personnel and contain the fire before it spreads uncontrollably. This urgency necessitates detecting fires with minimal delay, which often requires processing the data directly on edge devices to avoid potential delays caused by local or internet connections. To determine the most suitable backbone for deployment in such scenarios, we establish a weighted metric that considers three key factors:

- mAP@50 (Object detection accuracy), weight = 5  
Object detection accuracy is paramount for identifying fires accurately, especially in resource-constrained environments like edge computing in IoT applications. High accuracy ensures reliable detection.
- Model Size, weight = 3  
The model's size is critical in environments with limited resources, such as edge computing in IoT applications. Smaller models require less storage space and memory, making them more practical for deployment.
- FPS (Inference Speed), weight = 2  
While inference speed is essential for real-time applications, its significance might be slightly less than accuracy or model size in this context. Assigning moderate weight to FPS acknowledges its importance without overshadowing other critical factors.

To calculate the weighted scores for each model, we normalize (i.e. Norm) each parameter using min-max scaling to ensure balance across different units. The weighted score is computed as follows:

$$\text{Weighted Score} = (\text{Norm mAP@50} \times 5) + (\text{Norm Model Size} \times 1) + (\text{Norm FPS} \times 3) \quad (1)$$

Based on this calculation, the YOLOv5-CSP-Darknet53 (scale s) has the highest score, indicating its suitability for deployment in fire detection applications on edge devices.

Table 1. Backbone comparison

No	Backbone	mAP@50	Model Size (MB)	FPS	Weight Score
1	YOLOv5-CSP-Darknet53 (scale n)	0.773	3.9	96	3
2	YOLOv5-CSP-Darknet53 (scale s)	0.88	14.5	73	7.19
3	YOLOv5-CSP-Darknet53 (scale x)	0.837	173.2	29	4.28
4	YOLOv5-ResNet50	0.845	79.2	39	4.58
5	YOLOv5-ResNet101	0.83	117.48	29	3.70
6	YOLOv5-ResNet152	0.809	148.9	22	2.61
7	YOLOv5-MobileNetV3-Large	0.823	10.9	72	4.49
8	YOLOv5-MobileNetV3-Small	0.797	7.5	60	2.77
9	YOLOv5-ResNet 26 Layers	0.812	60	52	3.43
10	YOLOv5-ResNet 14 Layers	0.814	48.1	71	4.19
11	YOLOv5-ResNeXt50	0.843	78.2	35	4.35
12	YOLOv5-ResNeXt101	0.82	206.1	17	3.19

### 3.2. Comparison of fire and smoke detection speed with other methods

The comparative analysis depicted in the table highlights the performance metrics of distinct object detection methodologies, scrutinizing their mAP at an intersection over union (IoU) threshold of 0.50 (mAP@50) and corresponding frames per second (FPS) throughput as shown in Table 2. Li and Zhao [25] integrated YOLOv3 with the Darknet53 backbone, achieving a commendable mAP@50 score of 0.84. However, its speed is moderate at 24 FPS, indicating a trade-off between detection accuracy and real-time processing efficiency. Our investigation into a variant of the YOLO architecture employs YOLOv5 with CSP and the Darknet53 backbone, scaled to “s”. This adaptation attains a better mAP@50 score of 0.88. Because YOLOv5 with CSP is an advanced variant of the YOLO object detection framework, the cross stage partial network (CSPNet) is incorporated to enhance performance in several vital aspects. CSPNet divides the feature map of the base layer into two parts and merges them through a cross-stage hierarchy, reducing computational bottlenecks and enhancing gradient flow. This architecture leads to more efficient learning and better feature representation, allowing YOLOv5 to preserve gradient flow during backpropagation and improve the learning of complex patterns in data. As a result, the model achieves higher object detection accuracy, especially in scenarios with varied and intricate objects. Additionally, the CSP architecture optimizes computation by reusing feature maps, enabling the model to maintain high accuracy while being computationally efficient, crucial for real-time applications where processing speed is essential. Zheng *et al.* [26] propose an innovative Deformable-DETR-based fire detection framework, denoted as FTA-DETR, which achieves detection accuracy with a mAP@50 score of 0.88, similar to our method mAP@50. Remarkably, this method demonstrates a slightly faster throughput at 76 FPS compared to YOLOv5 with CSP, indicating a minor trade-off in throughput for maintaining high detection accuracy. Even though FTA-DETR exhibits a marginally higher FPS, the YOLOv5-CSP-Darknet53 (scale s) method still offers significantly better processing speed than the YOLOv3-Darknet53, with three times the FPS. This improvement underscores the enhanced computational efficiency and suitability of YOLOv5 with CSP for real-time applications where both speed and accuracy are critical.

Table 2. Performance comparison with existing methods

No	Authors	Method	mAP@50	FPS
1	Li and Zhao [25]	YOLOv3- Darknet53	0.84	24
2	Zheng <i>et al.</i> [26]	FTA-DETR	0.88	76
3	Our method	YOLOv5-CSP-Darknet53 (scale s)	0.88	73

### 3.3. Model deployment on website

The importance of this research lies in its potential to revolutionize public safety and industrial fire response systems. Fires are unpredictable and can escalate rapidly, leaving minimal time for human intervention. The YOLOv5 model, powered by the CSPDarkNet53 Scale-S backbone, brings a transformative advantage to these critical scenarios through its lightning-fast detection speed of 0.01 seconds. Fire detection simulations were conducted utilizing the Streamlit package for web-based applications. Testing was performed using video files, yielding results showcased in Figure 10. This ultra-rapid response capability ensures that even the earliest signs of fire or smoke are detected almost instantaneously, providing valuable seconds for emergency systems to act. These moments can make the difference between minor damage and catastrophic loss, potentially saving lives, property, and infrastructure.



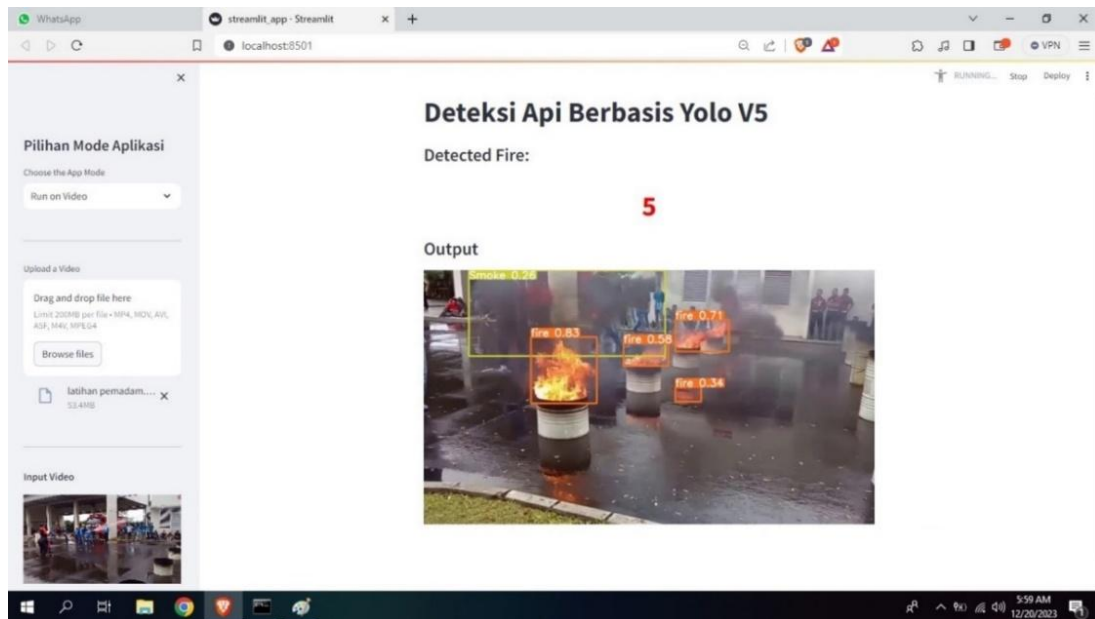


Figure 10. Running source video

Early detection is crucial for industrial environments, where fire hazards are often present due to the nature of equipment and materials. Traditional fire detection methods rely on sensors or manual observation, and they are both prone to delays and failures. Integrating YOLOv5-based fire detection systems into these environments can streamline safety protocols by automating real-time fire monitoring. This technology ensures that incidents are flagged immediately, triggering alarms, fire suppression systems, or emergency responses without waiting for human confirmation. Such efficiency can prevent production downtime and economic losses, enhancing the overall resilience of businesses.

The consequences of delayed fire detection can be severe in public spaces such as residential areas, schools, shopping centers, or hospitals where large numbers of people gather. By embedding the proposed technology into surveillance systems or standalone fire detection networks, authorities and emergency responders can be alerted within milliseconds of fire or smoke detection. It allows for faster evacuation procedures, targeted fire suppression efforts, and reduced risk of injuries. Ultimately, deploying this advanced detection system represents a paradigm shift in fire safety, replacing reactive approaches with proactive fire management to protect communities, critical infrastructure, and industrial operations more effectively than ever.

### 3.4. Limitations and implications of our study

The study highlights substantial fire and smoke detection progress using YOLO models, specifically the YOLOv5-CSP-Darknet53 (scale s) configuration. It combines high detection accuracy with efficient processing, making it a practical choice for real-time applications. Implemented in Streamlit, the model offers a user-friendly, web-based simulation environment for testing and deploying fire detection, enhancing accessibility for various users. However, our study identified several limitations affecting model deployment. First, the training dataset may contain biases, limiting the model's generalizability across diverse real-world scenarios emphasizing the need for varied datasets. Second, the model can generate false positives, particularly in low light or obstructed views, underscoring the need for improved robustness. Third, while the system performs in real-time, processing delays can occur with high-resolution video or on devices with limited resources, suggesting further optimization to ensure rapid response in critical situations.

Additionally, hardware limitations may restrict broader deployment, stressing the importance of compatibility across platforms for practical, widespread usage. Integrating CSPNet within the YOLO architecture enhances detection efficiency and accuracy, offering valuable insights into optimizing DNN configurations. Ultimately, this research underlines the potential of YOLO-based models to bolster safety measures across residential, commercial, and industrial settings while underscoring the importance of continued exploration into DNN architecture for further performance improvements.

#### 4. CONCLUSION

In conclusion, our research has successfully trained YOLOv5 models with various DNN configurations as backbones to detect fire and smoke in both videos and images. By integrating these trained models into the Streamlit package, which is accessible via web browsers, we achieved notable results. The highest mAP accuracy of 0.880 was attained using the YOLOv5 scheme with the CSPDarknet53 backbone, initially developed for YOLO. This achievement demonstrates the model's strong performance in detecting fire and smoke, highlighting its potential for practical applications. The implications of this research are significant, particularly in enhancing environmental and human safety by monitoring fire objects captured by CCTV cameras. Unlike conventional electronic-based sensors that activate only when smoke or fire reaches the sensor, our computer vision-based approach allows immediate detection of fire and smoke objects. This swift detection capability facilitates quicker response times and provides critical information for practical mitigation efforts, ultimately improving safety measures and response strategies. The future work will focus on several areas for improvement and expansion. Firstly, we aim to explore additional datasets to validate and generalize our models' performance across diverse scenarios and environmental conditions. Moreover, we plan to enhance our detection system's robustness and real-time capabilities, possibly by integrating advanced optimization techniques and parallel processing methodologies.

#### ACKNOWLEDGEMENTS

Through a fundamental research grant, this research has received support from Universitas Amikom Yogyakarta. The research grant number is RP-1725247579.

#### REFERENCES




- [1] K. Muhammad, J. Ahmad, and S. W. Baik, "Early fire detection using convolutional neural networks during surveillance for effective disaster management," *Neurocomputing*, vol. 288, pp. 30–42, 2018, doi: 10.1016/j.neucom.2017.04.083.
- [2] D. Qin, P. K. Gao, F. Aslam, M. Sufian, and H. Alabduljabbar, "A comprehensive review on fire damage assessment of reinforced concrete structures," *Case Studies in Construction Materials*, vol. 16, 2022, doi: 10.1016/j.cscm.2021.e00843.
- [3] S. Suryanto *et al.*, "Environmental impacts caused by fire and explosion accidents in maritime activities: A review," *E3S Web of Conferences*, vol. 563, 2024, doi: 10.1051/e3sconf/202456302035.
- [4] C. Li *et al.*, "Fast forest fire detection and segmentation application for UAV-assisted mobile edge computing system," *IEEE Internet of Things Journal*, vol. 11, no. 16, pp. 26690–26699, 2024, doi: 10.1109/JIOT.2023.3311950.
- [5] F. Khan, Z. Xu, J. Sun, F. M. Khan, A. Ahmed, and Y. Zhao, "Recent advances in sensors for fire detection," *Sensors*, vol. 22, no. 9, 2022, doi: 10.3390/s22093310.
- [6] J. Fonollosa, A. Solórzano, and S. Marco, "Chemical sensor systems and associated algorithms for fire detection: A review," *Sensors*, vol. 18, no. 2, 2018, doi: 10.3390/s18020553.
- [7] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik, "Convolutional neural networks based fire detection in surveillance videos," *IEEE Access*, vol. 6, pp. 18174–18183, 2018, doi: 10.1109/ACCESS.2018.2812835.
- [8] X. Deng, X. Shi, H. Wang, Q. Wang, J. Bao, and Z. Chen, "An indoor fire detection method based on multi-sensor fusion and a lightweight convolutional neural network," *Sensors*, vol. 23, no. 24, pp. 1–12, 2023, doi: 10.3390/s23249689.
- [9] J. Baek *et al.*, "Real-time fire detection system based on dynamic time warping of multichannel sensor networks," *Fire Safety Journal*, vol. 123, 2021, doi: 10.1016/j.firesaf.2021.103364.
- [10] L. Kiely *et al.*, "Assessing costs of Indonesian fires and the benefits of restoring peatland," *Nature Communications*, vol. 12, no. 1, pp. 1–11, 2021, doi: 10.1038/s41467-021-27353-x.
- [11] B. Pushpa and M. Kamarasan, "Video summarization based on Gaussian mixture model and kernel support vector machine for forest fire detection," *International Journal of Engineering and Advanced Technology*, vol. 9, no. 1, pp. 1827–1831, 2019, doi: 10.35940/ijeat.A1442.109119.
- [12] H. A. H. Mahmoud, A. H. Alharbi, and N. S. Alghamdi, "Time-efficient fire detection convolutional neural network coupled with transfer learning," *Intelligent Automation and Soft Computing*, vol. 31, no. 3, pp. 1393–1403, 2022, doi: 10.32604/IASC.2022.020629.
- [13] A. Bari, T. Saini, and A. Kumar, "Fire detection using deep transfer learning on surveillance videos," *Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2021*, pp. 1061–1067, 2021, doi: 10.1109/ICICV50876.2021.9388485.
- [14] X. Cheng, "Research on application of the feature transfer method based on Fast R-CNN in smoke image recognition," *Advances in Multimedia*, vol. 2021, 2021, doi: 10.1155/2021/6147860.
- [15] X. Zheng, F. Chen, L. Lou, P. Cheng, and Y. Huang, "Real-time detection of full-scale forest fire smoke based on deep convolution neural network," *Remote Sensing*, vol. 14, no. 3, 2022, doi: 10.3390/rs14030536.
- [16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.
- [17] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.
- [18] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv-Computer Science*, pp. 1–6, 2018.
- [19] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," *arXiv-Computer Science*, pp. 1–17, 2020.
- [20] G. Jocher, "Ultralytics YOLOv5," *Zenodo*, v7.0, 2020, doi: 10.5281/zenodo.3908559.
- [21] M. Karthi, V. Muthulakshmi, R. Priscilla, P. Praveen, and K. Vanisri, "Evolution of YOLO-V5 algorithm for object detection: automated detection of library books and performance validation of dataset," *2021 IEEE International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems*, pp. 1–6, 2021, doi: 10.1109/ICESSES2305.2021.9633834.






- [22] M. S. Nadeem, V. N. L. Franqueira, X. Zhai, and F. Kurugollu, "A survey of deep learning solutions for multimedia visual content analysis," *IEEE Access*, vol. 7, pp. 84003–84019, 2019, doi: 10.1109/ACCESS.2019.2924733.
- [23] A. Howard *et al.*, "Searching for mobileNetV3," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1314–1324, 2019, doi: 10.1109/ICCV.2019.00140.
- [24] A. Aslanhan, "FireV2 computer vision project," *Roboflow Universe*, 2023. Accessed: February 29, 2024. [Online]. Available: <https://universe.roboflow.com/ali-aslanhan/firev2-g6bie>
- [25] P. Li and W. Zhao, "Image fire detection algorithms based on convolutional neural networks," *Case Studies in Thermal Engineering*, vol. 19, Jun. 2020, doi: 10.1016/j.csite.2020.100625.
- [26] H. Zheng, G. Wang, D. Xiao, H. Liu, and X. Hu, "FTA-DETR: An efficient and precise fire detection framework based on an end-to-end architecture applicable to embedded platforms," *Expert Systems with Applications*, vol. 248, 2024, doi: 10.1016/j.eswa.2024.123394.

## BIOGRAPHIES OF AUTHORS






**Agung Nugroho**    earned a diploma in electrical engineering in 2011 at Gadjah Mada University, Yogyakarta, and a bachelor's degree in industrial engineering at Pattimura University, Ambon, Maluku, Indonesia in 2017. Currently pursuing a master's degree in information technology at Amikom University, Yogyakarta, Indonesia. Apart from that, he has the status of an employee of PT PLN (Persero) and serves as team leader for UP3 Banyuwangi Electricity System Planning, East Java, Indonesia. He can be contacted at email: [agung4b08@students.amikom.ac.id](mailto:agung4b08@students.amikom.ac.id).



**I Made Artha Agastya**    earned a bachelor's degree in aeronautical engineering, from Bandung Institute of Technology, Indonesia. The master's degree was obtained at the Master of Electrical Engineering, Gadjah Mada University, Yogyakarta, Indonesia. Doctoral education obtained from Taylor University Malaysia. Currently, he has the status as a lecturer in Informatics, at Amikom University, Yogyakarta, Indonesia. He can be contacted at email: [artha.agastya@amikom.ac.id](mailto:artha.agastya@amikom.ac.id).



**Kusrini**    earned a bachelor's to doctoral degree at Gadjah Mada University, Yogyakarta, Indonesia. Currently, he has the status of Professor and lecturer at the Master of Information Technology, Amikom University, Yogyakarta. She can be contacted at email: [kusrini@amikom.ac.id](mailto:kusrini@amikom.ac.id).