IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 August 2023, accepted 28 August 2023, date of publication 5 September 2023, date of current version 11 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3312217

RESEARCH ARTICLE

Assessing the Effectiveness of YOLO Architectures for Smoke and Wildfire Detection

EDMUNDO CASAS[1], (Member, IEEE), LEO RAMOS[1], (Student Member, IEEE), EDUARDO BENDEK[1,2], AND FRANCKLIN RIVAS-ECHEVERRÍA[1], (Senior Member, IEEE)

[1]Kauel Inc., Houston, TX 77027, USA
[2]Jet Propulsion Laboratory, NASA, Pasadena, CA 91109, USA

Corresponding author: Leo Ramos (leo.ramos@kauel.com)

ABSTRACT This paper presents a comprehensive evaluation of YOLO architectures for smoke and wildfire detection, including YOLOv5, YOLOv6, YOLOv7, YOLOv8, and YOLO-NAS. We aim to assess their effectiveness in early detection of wildfires. The Foggia dataset is used for this, and performance metrics such as Recall, Precision, F1-score, and mean Average Precision are employed. Our methodology trains each architecture for 300 epochs, focusing on recall for its relevance in this area. The 'best models' are evaluated on the Foggia test set and further tested with a challenging, custom-assembled dataset from independent online sources to assess real-world performance. Results show that YOLOv5, YOLOv7, and YOLOv8 exhibit a balanced performance across all metrics in both validation and testing. YOLOv6 performs slightly lower in recall during validation but achieves a good balance on testing. YOLO-NAS variants excel in recall, making them suitable for minimizing missed detections. However, precision performance is lower for YOLO-NAS models. Visual results demonstrate that top-performing models accurately identify most instances in the test set. However, they struggle with distant scenes and poor lighting conditions, occasionally detecting false positives. In favorable conditions, the models perform well in identifying relevant instances. We conclude that no single model excels in all aspects of smoke and wildfire detection. The choice of model depends on specific application requirements, considering accuracy, recall, and inference time. This research enriches the field of computer vision for smoke and wildfire detection, laying a foundation for system enhancements and serving as a basis for future research to optimize detection effectiveness.

INDEX TERMS Artificial intelligence, computer vision, deep learning, neural networks, object detection, smoke, wildfire, YOLO.

I. INTRODUCTION

Wildfires pose a significant threat to ecosystems, human lives, and infrastructure, making their early detection and mitigation crucial [1]. These devastating events, characterized by uncontrolled fires that rapidly spread across forested areas, have severe consequences for biodiversity, air quality, and the safety of communities [2], [3], [4]. The intensity and frequency of wildfires have escalated in recent years, particularly in regions with dense forest coverage and dry climates [5], [6].

In many parts of the world, such as California in the United States, Australia, the Amazon rainforest, and the Mediterranean basin, wildfires have become increasingly common and destructive [3], [7]. These regions have experienced large-scale infernos that ravage vast stretches of forestland, endangering wildlife, causing significant economic losses, and displacing local populations. The impacts of climate change, coupled with human activities such as land clearance and improper fire management practices, have contributed to the escalating wildfire problem on a global scale [8].

Traditional methods of wildfire detection and monitoring predominantly rely on human observation and reporting, often resulting in delays in response and allowing fires to grow unchecked [9]. As a result, there is an urgent need for advanced technologies that can enable early detection and provide real-time situational awareness to facilitate prompt and effective firefighting efforts [10].

Recent advancements in computer vision and artificial intelligence have offered promising solutions for smoke and

The associate editor coordinating the review of this manuscript and approving it for publication was Okyay Kaynak.

wildfire detection [11]. Object detection algorithms, in particular, have demonstrated their potential in automatically identifying and localizing fire-related objects, such as smoke plumes and flames, in images and videos [12]. Among these algorithms, the YOLO (You Only Look Once) architecture has emerged as one of the most efficient and accurate frameworks for object detection tasks [13].

This paper presents a comprehensive performance evaluation of several YOLO architectures, including YOLOv5, YOLOv6, YOLOv7, YOLOv8, and YOLO-NAS, for smoke and wildfire detection. Specifically, YOLOv8 and YOLO-NAS are the two most recent YOLO architectures released. These recent advancements represent the forefront of object detection in the YOLO framework and provide an exciting opportunity to evaluate their potential for smoke and wildfire detection. By examining these state-of-the-art architectures, we aim to gain insights into their unique features and assess their effectiveness in addressing the challenges posed by smoke and wildfire detection.

To assess the performance of the evaluated architectures, our study employs a set of metrics, including Recall, Precision, F1-score, and mean Average Precision (mAP). These metrics provide a holistic evaluation of the models' performance in terms of their ability to detect smoke and wildfire instances accurately. Additionally, our research utilizes the Foggia dataset, a dataset specifically designed for smoke and wildfire detection.

Our rigorous methodology involves training each architecture for a fixed 300 epochs, focusing on maximizing recall for its application in wildfire and smoke detection. The best-performing models are first evaluated using the Foggia dataset's test set for comparative analysis. Subsequently, these top models are further challenged using a custom-assembled unbiased test dataset from various independent online sources to assess their real-world applicability. Through this, we aim to provide a comprehensive understanding of the strengths and limitations of each YOLO architecture in the context of smoke and wildfire detection.

The findings of this study offer valuable insights into the strengths and limitations of these architectures, as we explore their varying abilities in aspects such as reducing false positives, increasing true positives, and more. This nuanced evaluation serves as a roadmap for developing more reliable and effective wildfire detection systems, enabling improved preparedness, timely response, and enhanced protection of ecosystems and communities at risk. Furthermore, our findings guide the specialized application or development of these architectures based on specific requirements, extending their relevance to other areas beyond wildfire detection.

II. METHODS

A. DATASET

In this study, we harness the power of the Foggia dataset,[1] specifically designed for detecting smoke and wildfires. Foggia was updated last in February 2023, keeping it up-to-date and pertinent to the evolving demands of wildfire detection.

The Foggia dataset is an assemblage of 8,974 diverse images, specifically designed to aid in the detection of wildfires and smoke. Out of these, 3,731 images depict fire, while 6,791 images capture the appearance of smoke, as demonstrated in Fig. 1.

To facilitate effective learning, a substantial portion of this dataset, 70% or around 6,300 images, is used for model training. The wide variety of images expose the models to diverse scenarios, thereby fostering comprehensive learning. Model performance during training is monitored using about 20% of the data, approximately 1,800 images, forming the validation set. This helps in identifying the most efficient model. Lastly, a collection of 899 images, forming the remaining 10% of the data, is set aside for testing. These images provide a neutral ground to measure the model's generalization capabilities and evaluate its efficacy in detecting smoke and wildfires in real-world scenarios.

B. ARCHITECTURES UNDER STUDY

1) YOLOv5

YOLOv5 [14], or "You Only Look Once version 5," is an innovative object detection algorithm renowned for its reliability, high accuracy, and remarkable simplicity. Released by Ultralytics[2] in June 2020, YOLOv5 has rapidly established itself as a leading solution in the object detection field, attributed to its continuous evolutionary iterations that have optimized both performance and speed [15].

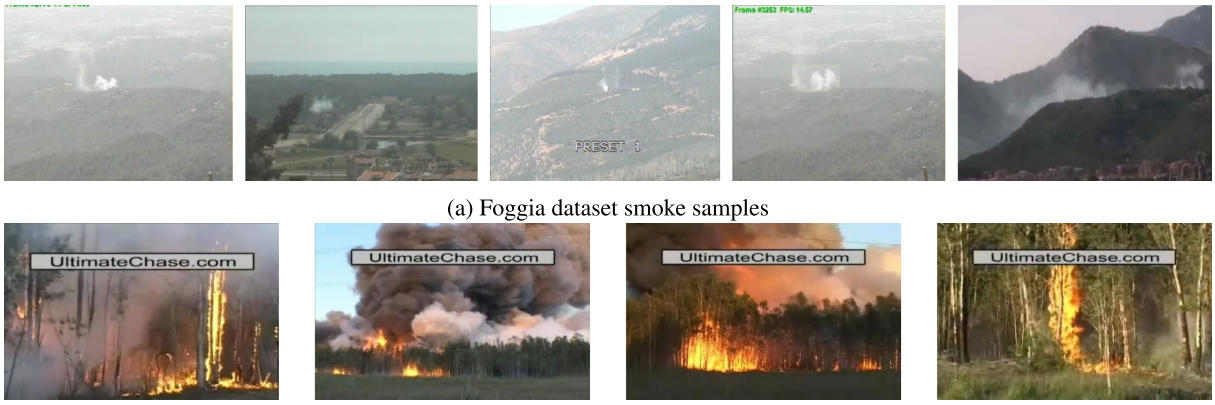
At its core, the YOLOv5 model is composed of three integral parts: the backbone, the neck, and the head, as seen in Fig. 2. The backbone of YOLOv5 employs the Cross Stage Partial (CSP) network strategy within the CSP-Darknet53 convolutional network [16]. This strategy ensures a robust flow of information, especially in deep layers, and effectively mitigates issues related to redundant gradients and vanishing gradients. The information-rich features extracted by the backbone from the input image greatly enhance the model's ability to detect objects across various scales [17].

The neck of the YOLOv5 model incorporates a variation of the Spatial Pyramid Pooling (SPP) and integrates the BottleNeckCSP into the Path Aggregation Network (PANet) [18]. This fusion of techniques helps increase the receptive field, isolating crucial context features without compromising the network speed. The PANet, originally a feature pyramid network used in YOLOv4, is further refined in YOLOv5 by the CSPNet strategy [17], offering improved precision in pixel localization. This part of the architecture is particularly critical in handling object scaling and enabling the model to perform exceptionally well on unseen data.

The final component, the head of the YOLOv5 model, remains consistent with its predecessors, comprising three convolution layers [17]. These layers predict bounding box

[1] https://universe.roboflow.com/fire-detection-uoeha/foggia-all

[2] https://ultralytics.com/



(a) Foggia dataset smoke samples

(b) Foggia dataset fire samples

FIGURE 1. Sample images from the Foggia dataset.

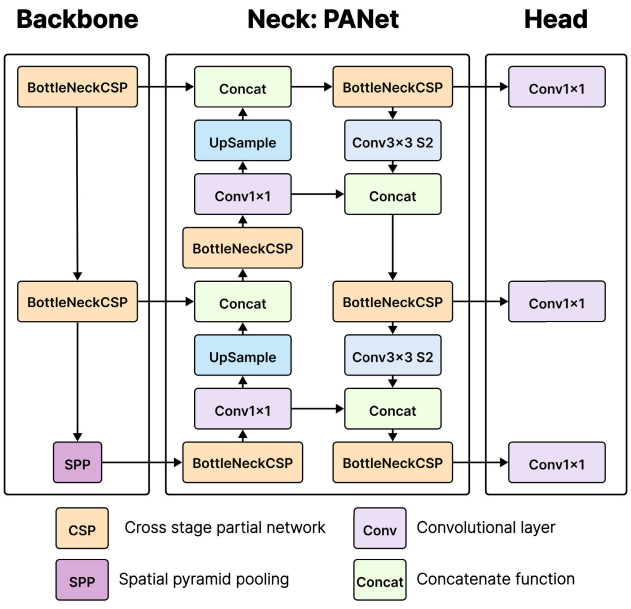


FIGURE 2. YOLOv5 architecture structure.

coordinates, scores, and object classes, featuring a minor alteration from the preceding versions in the computation of target coordinates for bounding boxes [17], [19].

Importantly, YOLOv5 offers different variants to suit diverse needs and contexts. In this paper, we will investigate five such variants: nano (n), small (s), medium (m), large (l), and extra large (x), details of which can be found in Table 1.

2) YOLOv6

YOLOv6, standing for "You Only Look Once version 6," is an object detection architecture striving to harmonize speed and accuracy through innovative methodologies [20]. It was was designed by a team from Meituan,[3] a Chinese e-commerce platform company, which is why it is also known as Meituan-YOLOv6 and MT-YOLOv6. As seen in Fig. 3, this model remains faithful to the tripartite structure of its forebears, consisting of the Backbone, Neck, and Head. However, YOLOv6 distinguishes itself by introducing an anchor-free model with a reparameterized backbone, enhancing its uniqueness in the field [20].

[3]https://www.meituan.com/

TABLE 1. Details of the studied YOLOv5 variants.

Model	Size (pixels)	Params (M)	FLOPs (B)
YOLOv5n	640	1.9	4.5
YOLOv5s	640	7.2	16.5
YOLOv5m	640	21.2	49
YOLOv5l	640	46.5	109.1
YOLOv5x	640	86.7	205.7

At the heart of the YOLOv6 architecture lies its backbone, which holds a pivotal role in feature extraction. Extracted features feed into the network's neck and head sections, shouldering the majority of the computational load. To reconcile the opposing requirements of speed and accuracy often encountered in traditional multi-branch networks like ResNets and linear networks like VGG, YOLOv6 introduces reparameterized backbones [21]. This technique adjusts the network structure during training and inference.

The smaller YOLOv6 models (nano, tiny, and small) harness reparameterized VGG networks (RepBlock) with skip connections for training, which transition to simple 3×3 convolutional (RepConv) blocks during inference [22]. Meanwhile, the medium and large YOLOv6 models deploy reparameterized versions of the CSP backbone, termed CSPStackRep, culminating in the EfficientRep backbone [21], [23].

The neck of YOLOv6, akin to other object detection models, harvests multi-scale feature maps using Path Aggregation Networks (PAN) [24]. The innovative Rep-PAN in YOLOv6 amalgamates features from a range of

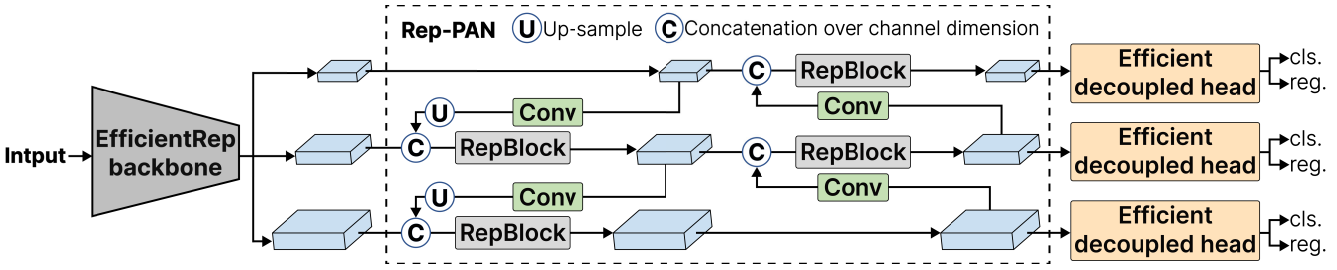


FIGURE 3. YOLOv6 architecture structure.

TABLE 2. Details of the studied YOLOv6 variants.

Model	Size (pixels)	Params (M)	FLOPs (G)
YOLOv6n	640	4.7	11.4
YOLOv6s	640	18.5	45.3
YOLOv6m	640	34.9	85.8
YOLOv6l	640	59.6	150.7

reparameterized blocks, bolstering its capacity for hardware-friendly design [20].

In a significant departure from its predecessors, YOLOv6 debuts the Efficient Decoupled Head [21]. This unique architecture ensures that the classification and detection branches no longer share parameters, branching off independently from the backbone, which effectively reduces computational requirements while amplifying accuracy [20], [21].

Finally, YOLOv6 leverages two distinct loss functions: Varifocal Loss (VFL) for classification and Distribution Focal Loss (dfl) in tandem with either SIoU or GIoU for box regression [25]. VFL, a derivative of focal loss, effectively manages both challenging and easy examples during training by assigning differential weights and attributing varying degrees of importance to positive and negative examples [20]. This approach promotes balanced learning signals from both sample types. Medium and large YOLOv6 models apply dfl for box regression loss, treating the continuous distribution of box locations as a discretized probability distribution, which proves particularly potent in scenarios with unclear ground truth boundaries [20], [26]. These synergistic components culminate to define YOLOv6 as a potent contender in the realm of object detection algorithms.

In this paper, we will delve into four different variants of YOLOv6: nano (n), small (s), medium (m), and large (l), as shown in Table 2. Each of these variants presents a unique combination of features and parameters, contributing to the overall versatility and adaptability of YOLOv6 as an object detection model.

3) YOLOv7

YOLOv7 [27] (You Only Look Once version 7) represents a significant advancement in the realm of object detection models, following the established YOLO framework with its three main components: the backbone, neck, and head, as shown in Fig. 4. Each of these components plays a crucial role in the process of image recognition and object detection. The aspiration behind the development of YOLOv7 was to design a network architecture capable of predicting bounding boxes with superior accuracy when compared to similar models, while maintaining comparable inference speeds [27], [28]. This aspiration led to several significant modifications in the YOLO network and its training procedures, further optimizing each component of the YOLO framework.

One of the key enhancements introduced in YOLOv7 is the Extended Efficient Layer Aggregation Network (E-ELAN), an extended version of the ELAN computational block [29]. This modification enhances the efficiency of the backbone's convolutional layers within the YOLO networks, an element essential for maintaining efficient inference speed. It considers memory requirements for layer retention and the distance the gradient has to back-propagate through the layers, with an aim for a shorter gradient to enhance the network's learning capabilities [27], [29].

A novel inclusion in the network is the re-parameterization planning [30]. This process averages a set of model weights to form a more robust model for the patterns it is modeling. The authors used gradient flow propagation paths to determine which network modules should incorporate these strategies. Furthermore, recognizing that different applications require varying levels of accuracy and inference speeds, YOLOv7 adopts model scaling techniques. In this technique, the network's depth and width are scaled concurrently while concatenating layers together, helping to maintain an optimal model architecture when scaling for different sizes [27].

YOLOv7 also introduces the Auxiliary Head Coarse-to-Fine concept. An auxiliary head, added in the middle of the network, is supervised during training alongside the head making the actual predictions [31]. Given its proximity to the prediction, the auxiliary head does not train as efficiently as the final head, which led the authors to experiment with different levels of supervision for this head, settling on a coarse-to-fine definition where supervision is passed back from the lead head at varying granularities [27], [31].

YOLOv7 and its variants, depart from using ImageNet pre-trained backbones. Instead, they rely entirely on the COCO

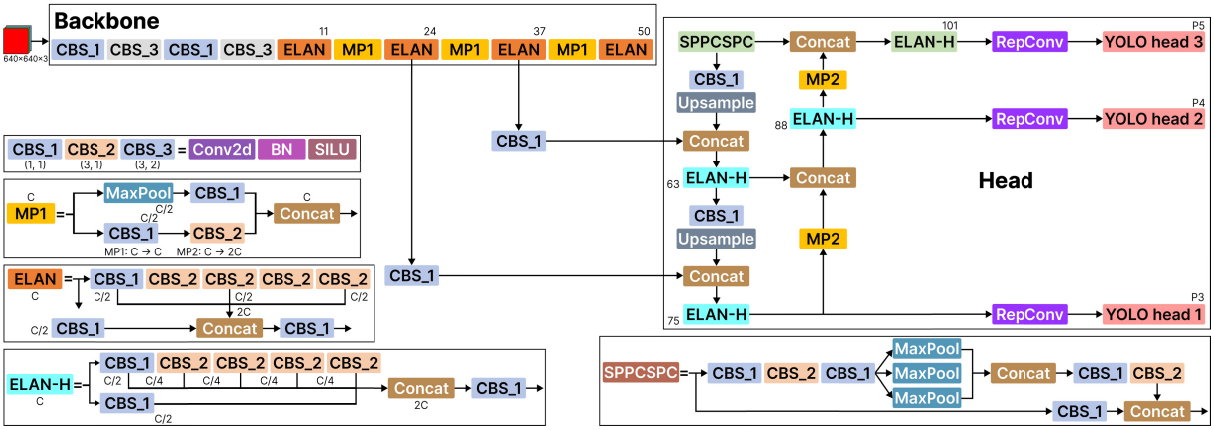


FIGURE 4. YOLOv7 architecture structure.

TABLE 3. Details of the studied YOLOv7 variants.

Model	Size (pixels)	Params (M)	FLOPs (G)
YOLOv7	640	36.9	104.7
YOLOv7x	640	71.3	189.9

dataset for training. In the context of this paper, our focus will be on YOLOv7 (base version) and YOLOv7x. Detailed characteristics of these versions can be found in Table 3.

The innovative modifications and enhancements introduced in YOLOv7 marked significant contributions to the field of computer vision research. With improved speed and accuracy, architectural reforms, model scaling, and a unique coarse-to-fine approach, YOLOv7 standed as a noteworthy advancement in object detection capabilities.

4) YOLOv8

YOLOv8 [32], the latest addition to the influential 'You Only Look Once' series, was introduced by Ultralytics on January 10th, 2023. Capitalizing on the groundwork laid by the successful YOLOv5 model, YOLOv8 signifies a major advancement in the realm of object detection, image classification, and instance segmentation [33]. Despite the absence of a published paper, the available repository and community discussions provide significant insights into the revolutionary changes introduced with YOLOv8.

Similar to its predecessors, YOLOv8 adheres to the three main architectural components that define the YOLO framework: the Backbone, Neck, and Head. These components work in unison, with the Backbone featurizing image frames, the Neck combining these features, and the Head executing the prediction of object locations and classes, as illustrated in Figure 5.

One of the salient features of YOLOv8 is the adoption of an anchor-free model, which departs from the anchor-box approach found in earlier YOLO models [34]. This innovative shift permits the model to predict an object's center directly, mitigating challenges associated with anchor boxes, such as lack of generalization and difficulty in handling irregularities. By reducing the number of box predictions, YOLOv8 enhances the speed of the Non-Maximum Suppression (NMS) process, a crucial post-processing step responsible for sifting through candidate detections after inference [30], [33].

YOLOv8 also debuts architectural refinements related to convolutions, the fundamental building blocks of neural networks. The introduction of C2f, replacing C3, along with the substitution of the initial 6×6 convolution in the stem with a 3×3 convolution, has ushered in a more efficient and flexible model structure [33], [35]. In this new design, all outputs from the Bottleneck, which consists of two 3×3 convolutions with residual connections, are concatenated, contrasting the C3 setup where only the output of the last Bottleneck was utilized [36]. This adjustment ensures a more robust model structure, pushing the boundaries of YOLOv8's capabilities in computer vision tasks.

In a continuous effort to cater to objects of various scales, YOLOv8 incorporates the Spatial Pyramid Pooling Feature (SPPF) [30], [37]. Furthermore, this model utilizes an online image augmentation strategy during training, such as the mosaic augmentation. This technique, which involves stitching four images together, allows the model to learn objects in new locations, in partial occlusion, and against varying surrounding pixels [38].

YOLOv8 serves as a remarkable advancement in the YOLO series, further elevating the standards for future computer vision projects. Through high accuracy rates, innovative architectural changes, and superior developer features, YOLOv8 continues to evolve, becoming an increasingly appealing choice in the realm of real-time object detection.

Recognizing the diverse needs of different applications, YOLOv8, like its predecessors, comes in various versions: nano (n), small (s), medium (m), large (l), and extra large (x).

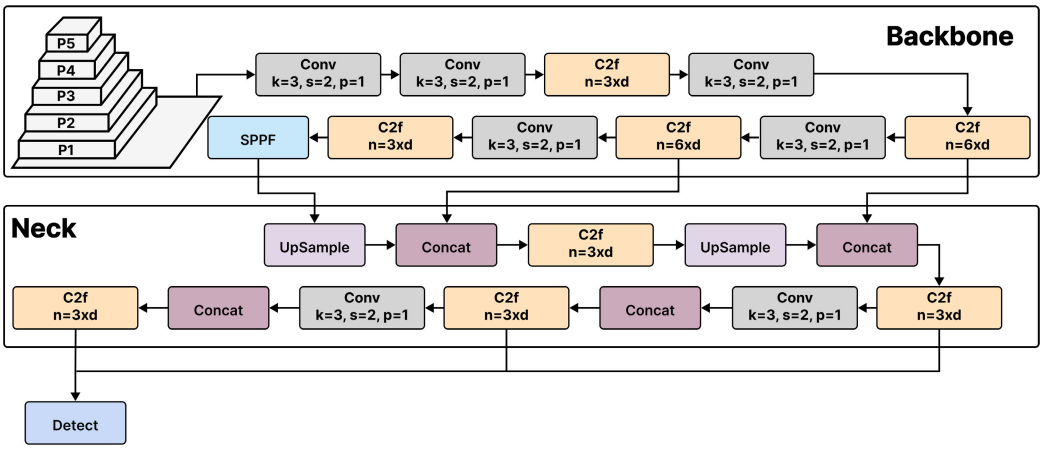


FIGURE 5. YOLOv8 architecture structure.

TABLE 4. Details of the studied YOLOv8 variants.

Model	Size (pixels)	Params (M)	FLOPs (B)
YOLOv8n	640	3.2	8.7
YOLOv8s	640	11.2	28.6
YOLOv8m	640	25.9	78.9
YOLOv8l	640	43.7	165.2
YOLOv8x	640	68.2	257.8

We will delve into an in-depth analysis of these five versions, the details of which can be found in Table 4.

5) YOLO-NAS

YOLO-NAS is an innovative real-time object detection model, the brainchild of Deci.ai,[4] which capitalizes on the most recent advancements in deep learning technology. It marks a significant milestone in the YOLO series, addressing key limitations of previous YOLO models and propelling the capabilities of real-time object detection to new heights.

The term 'NAS' in YOLO-NAS represents 'Neural Architecture Search' [39]. Unlike conventional methods that depend heavily on human insight and manual design, NAS utilizes optimization algorithms to automate the process of designing neural network architectures. The primary goal of NAS is to achieve an optimal balance between model accuracy, computational complexity, and model size [39], [40]. A high-level overview of the architecture of YOLO-NAS can be seen in Fig. 6. Also, it is worth mentioning that YOLO-NAS is equipped with three model architectures compatible with different precisions: FP32 (single precision floating point), FP16 (half precision floating point), and INT8 (8-bit integer).

The architectural design of YOLO-NAS models is primarily driven by Deci's proprietary NAS technology, AutoNAC. This specialized engine streamlines the optimization of sizes and structures for various stages, inclusive of block type, number of blocks, and the number of channels per stage [28]. Within the NAS search space, AutoNAC evaluated 1014 potential architecture configurations, bearing in mind all components of the inference stack, including compilers and quantization. This comprehensive assessment enabled AutoNAC to identify an 'efficiency frontier' within the search space, representing the region housing the best-performing models. All three YOLO-NAS models are products of this optimal region.

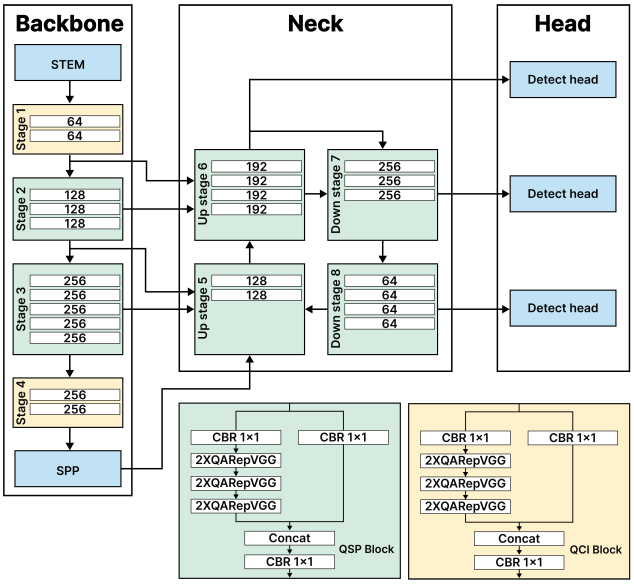


FIGURE 6. High-level overview of YOLO-NAS architecture.

A notable feature of YOLO-NAS is the integration of Quantization-Aware RepVGG (QA-RepVGG) blocks into the model's architecture [41]. These blocks guarantee compatibility with Post-Training Quantization (PTQ) [42], thus minimizing accuracy loss during this process. The model's distinct 'QSP' and 'QCI' modules, comprised of QA-RepVGG blocks, support 8-bit quantization and reparameterization. Moreover, Deci's team implemented a hybrid

[4]https://www.supergradients.com/

quantization technique, selectively applying quantization to specific layers to optimize the tradeoff between accuracy and latency while preserving overall performance.

Furthermore, YOLO-NAS models harness the power of attention mechanisms and inference time reparameterization to bolster their object detection prowess. All the details of the creation of YOLO-NAS can be found on the official Deci[5] website. Considering the variety of potential use cases, YOLO-NAS is offered in several versions: small (s), medium (m), and large (l). An in-depth examination of these variants will be presented, with specific details illustrated in Table 5.

TABLE 5. Details of the studied YOLO-NAS variants.

Model	Size (pixels)	Params (M)
YOLO-NASs	640	19
YOLO-NASm	640	51.1
YOLO-NASl	640	66.9

Even without an official paper published, the development of YOLO-NAS signals a considerable evolution in the field of object detection. Its exemplary performance in terms of mAP and inference latency metrics indicates its potential applicability to high-demand, real-time detection tasks. As further information about its training regimen becomes available, it is anticipated that the full power and potential of this novel model will be better understood. In sum, YOLO-NAS embodies the cutting edge of real-time object detection, bringing a new dimension to the YOLO series with its advanced automated architecture design approach.

C. TRAINING AND EVALUATION

To ensure a comprehensive and equitable assessment of various deep learning architectures for object detection tasks, it is paramount to utilize a systematic comparison methodology. In this paper, we propose a rigorous training and evaluation method, particularly designed to examine the performance of the deep learning architectures we've studied. This method incorporates a fixed number of training epochs, followed by an evaluation of the chosen 'best model' on the test set.

Initially, each architecture undergoes training for an exact number of epochs, specifically set at 300. This number enables ample exploration of the models' learning abilities over a significant period. During this training process, the models' performances are routinely evaluated on a separate validation dataset to track their progress.

Upon completing the 300 epochs, the 'best model' is selected based on its performance regarding recall (see II-D) on the validation set. This choice is primarily influenced by the context of our study – wildfire and smoke detection. In this scenario, the emphasis is on minimizing false negatives, which could have severe consequences in real-world applications. Thus, recall, the metric that quantifies the model's ability to identify all relevant instances, becomes paramount in determining the 'best model'.

Subsequently, the selected best models from each architecture are evaluated on the dedicated test set. This evaluation serves to gauge their generalization capabilities and real-world performance, yielding insights into their effectiveness in smoke and wildfire detection scenarios.

By adhering to this systematic methodology, our aim is to establish a solid foundation for comparing the performance of different deep learning architectures in object detection tasks. This method ensures a meticulous evaluation process, offering a comprehensive understanding of the relative performance of the studied models. It facilitates informed decision-making and meaningful conclusions for future research and practical applications in the vital field of wildfire detection. The overall workflow process followed is illustrated in Fig. 7.

D. PERFORMANCE METRICS

To evaluate the performance of all models and enable effective comparisons, we have selected several commonly used metrics from the literature. These metrics provide an objective measure of the accuracy and efficiency with which the models can detect both smoke and wildfires. Next, we provide a detailed description of each selected metric.

1) PRECISION

Precision is a fundamental metric used in object detection tasks to assess the accuracy of the model's positive predictions. It quantifies the proportion of correctly identified positive instances out of all instances predicted as positive [43].

In mathematical terms, the definition of precision can be represented by the formulation presented in Equation 1, where TP represents the number of correctly predicted positive instances. False Positives FP represents the number of instances falsely predicted as positive.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

In the context of object detection, precision evaluates the model's ability to precisely locate and classify objects. It helps determine the reliability of the model in identifying positive instances, thus minimizing false positives [43]. A higher precision indicates a lower rate of falsely predicted positive instances.

2) RECALL

Recall, also known as the true positive rate or sensitivity, measures the proportion of actual positive instances correctly identified by the model [43], [44]. In the context of object detection, recall quantifies the model's ability to correctly detect and capture instances of objects.

Mathematically, it is defined as shown in Equation 2, where TP represents the number of correctly predicted positive instances, and FN represents the number of instances falsely

[5]https://deci.ai/blog/yolo-nas-object-detection-foundation-model/

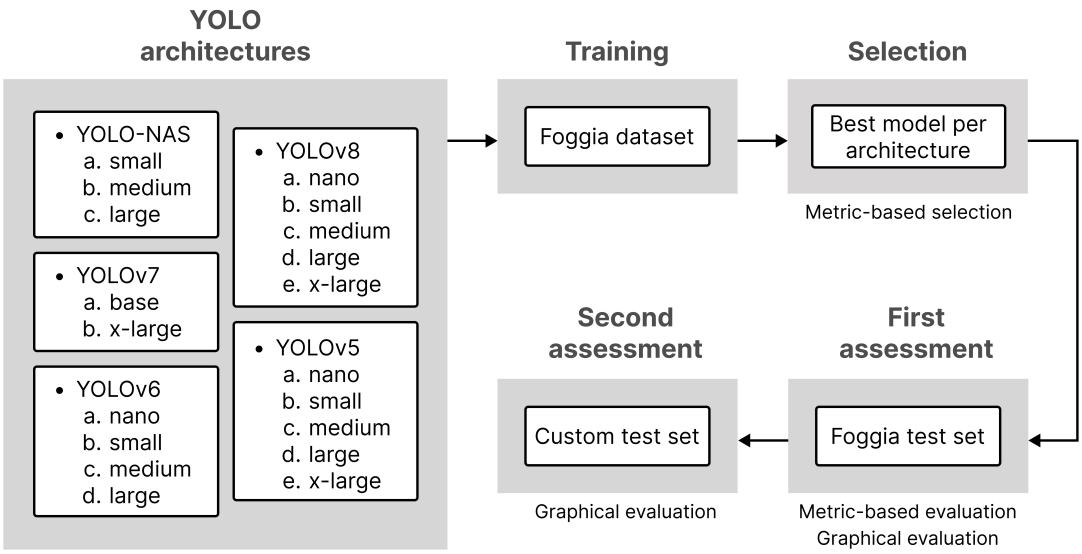


FIGURE 7. Workflow followed in this work.

predicted as negative.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

A higher recall indicates a lower rate of missed detections, which is desirable in applications where false negatives can have severe consequences.

3) F1-SCORE

The F1-score is a balanced metric that combines both precision and recall into a single value. It calculates the harmonic mean of precision and recall, providing an overall measure of the model's accuracy in object detection [45].

Mathematically speaking, its definition is expressed by the equation shown as Equation 3, where *Precision* is the ratio of true positives to the sum of true positives and false positives, and *Recall* is the ratio of true positives to the sum of true positives and false negatives.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

The F1-score is particularly useful when the dataset is imbalanced or when there is an uneven cost associated with false positives and false negatives. A higher F1-score indicates a better balance between precision and recall, showcasing the model's ability to accurately detect objects while minimizing both false positives and false negatives [43], [45].

4) MEAN AVERAGE PRECISION

Mean Average Precision (mAP) is a commonly used evaluation metric for object detection. It quantifies the precision and recall trade-off by calculating the average precision (AP) for each class and then taking the mean across all classes [46]. AP measures the precision at different levels of recall by computing the area under the precision-recall curve [43], [46].

Expressed mathematically, it is defined as depicted in Equation 4, where *precision*(r) represents the precision at a given recall level (r). A higher mAP indicates better performance in object detection, considering both precision and recall.

$$\text{AP} = \int_0^1 \text{precision}(r)\, \text{d}r \tag{4}$$

There are specific variations of mAP that focus on different levels of IoU (Intersection over Union) thresholds. mAP@0.5 measures the mean average precision at an IoU threshold of 0.5, while mAP@0.5-0.95 represents the mean average precision across a range of IoU thresholds from 0.5 to 0.95. In this work, we focus on evaluating the performance of the deep learning architectures under study using mAP@0.5.

5) LOSS

Loss refers to a quantitative measure of the discrepancy between the predicted outputs of a model and the actual or expected values. In the context of object detection, the loss is used to evaluate the accuracy of the detections made by the model compared to the real object locations and labels in an image. A lower loss value indicates a higher agreement between predictions and actual values, and the goal is to minimize the loss during the training process to improve the model's performance.

Different versions of YOLO (You Only Look Once) differ in the loss functions they utilize. YOLOv5 employs box loss, objectness (obj) loss, and classification (cls) loss. YOLOv6 utilizes intersection over union (IoU) loss, cls loss, and distributional focal loss (dfl). YOLOv7 employs box loss, obj loss, and cls loss. YOLOv8 utilizes box loss, cls loss, and dfl. Lastly, YOLO-NAS employs IoU loss, cls loss, and dfl. We provide a brief description of each of these losses below.

- **Box loss:** This loss function measures the discrepancy between predicted bounding box coordinates and the ground truth box coordinates [47]. It typically utilizes metrics such as mean squared error (MSE) or smooth L1 loss.
- **Objectness (obj) loss:** The obj loss evaluates the accuracy of objectness predictions, which determine whether a given bounding box contains an object or not [48]. It commonly employs binary cross-entropy loss.
- **Classification (cls) loss:** This loss measures the disparity between predicted class probabilities and the true class labels [49]. It utilizes categorical cross-entropy loss.
- **Intersection over Union (IoU) loss**: The IoU loss assesses the consistency between predicted bounding boxes and the ground truth boxes using the IoU metric [50]. It quantifies the overlap between the predicted and true bounding boxes.
- **Distributional Focal Loss (dfl):** This loss function is an extension of the focal loss and is used to handle class imbalance in object detection [51]. It assigns higher weights to challenging examples that are harder to classify accurately.

E. IMPLEMENTATION ENVIRONMENT

We conducted the models training on a high-performance computing (HPC) cluster using two Nvidia A100 SXM4 40GB GPUs, 32 CPU cores, and 64GB of system memory. We maintained all the original hyperparameters for each model, as our objective is to establish a baseline for comparison rather than achieving the best possible results by making modifications or incorporating additional techniques. The batch size used for training was set to 64.

To ensure consistency in evaluation, we made necessary adjustments to the code to consider recall as the primary metric for determining the 'best model.' This choice is particularly relevant to our field of application, which focuses on wildfire and smoke detection. Recall provides valuable insights into the model's ability to detect and capture instances of interest accurately.

All the code implementations were developed in Python programming language, and we obtained them from official repositories, ensuring reliability and adherence to standard practices in the field.

III. RESULTS AND DISCUSSION

A. YOLOv5

Table 6 presents the training time required for each variant of YOLOv5. It is important to note that the training time is measured in hours. As observed, the training time increases as we move from YOLOv5n to YOLOv5x, with YOLOv5n being the fastest to train and YOLOv5x requiring the most time. These results align with our expectations, as YOLOv5x is a larger and more complex model compared to YOLOv5n.

TABLE 6. Training time of each YOLOv5 variant.

Model	Training time (hours)
YOLOv5n	**1.512**
YOLOv5s	1.590
YOLOv5m	2.157
YOLOv5l	2.860
YOLOv5x	4.159

These results provide insights into the computational cost associated with training each variant. It is important to note that training time is one aspect to consider when selecting a model, but it should be evaluated alongside other performance metrics and inference speed.

Table 7 displays the performance metrics of each YOLOv5 variant on the validation dataset.

When comparing the precision values, YOLOv5s demonstrates the highest precision of 0.908, indicating its ability to minimize false positive detections. It is followed closely by YOLOv5m and YOLOv5n, which achieve precision values of 0.901. YOLOv5l and YOLOv5x exhibit slightly lower precision values of 0.893 and 0.889, respectively.

Moving on to recall, YOLOv5s achieves the highest value of 0.891, indicating its effectiveness in capturing actual instances of wildfires and smoke. YOLOv5n closely follows with a recall of 0.883, showing its capability in detecting these instances. YOLOv5m, YOLOv5x, and YOLOv5l also perform well, with recall values of 0.889, 0.885, and 0.882, respectively.

Analyzing the F1-score, YOLOv5s stands out with the highest score of 0.899, showcasing a good balance between precision and recall. YOLOv5n follows closely with an F1-score of 0.891, indicating its ability to achieve a trade-off between precision and recall. YOLOv5m, YOLOv5l, and YOLOv5x achieve F1-scores of 0.895, 0.887, and 0.887, respectively, demonstrating their competitive performance in capturing both precision and recall.

Considering the mAP@50 metric, YOLOv5n achieves the highest score of 0.920, closely followed by YOLOv5s with a score of 0.919. YOLOv5m, YOLOv5l, and YOLOv5x achieve mAP@50 scores of 0.910, 0.905, and 0.902, respectively.

These results suggest that YOLOv5s and YOLOv5n excel in terms of precision, recall, F1-score, and mAP@50, making them promising choices for our wildfire and smoke detection application. However, the other variants, YOLOv5m, YOLOv5l, and YOLOv5x, also exhibit competitive performance in terms of precision, F1-score, and mAP@50, demonstrating their potential for wildfire and smoke detection.

Figure 8 presents the evolution charts of all metrics for each YOLOv5 variant on the validation set. It can be observed that the YOLOv5n model exhibits slower convergence across all metrics compared to the other variants. The YOLOv5s model also demonstrates slower convergence, particularly in terms of recall. On the other hand, the YOLOv5m, YOLOv5l,

TABLE 7. Performance metrics for each YOLOv5 variant on validation.

Model	Precision	Recall	F1-score	mAP@50
YOLOv5n	0.901	0.883	0.891	**0.920**
YOLOv5s	**0.908**	**0.891**	**0.899**	0.919
YOLOv5m	0.901	0.889	0.895	0.910
YOLOv5l	0.893	0.882	0.887	0.905
YOLOv5x	0.889	0.885	0.887	0.902

TABLE 8. Best recall and corresponding epoch for each YOLOv5 Variant on validation.

Model	Best recall	Epoch
YOLOv5n	0.901	175
YOLOv5s	**0.908**	226
YOLOv5m	0.905	126
YOLOv5l	0.901	132
YOLOv5x	0.904	131

and YOLOv5x variants show faster convergence and achieve stable performance in fewer iterations. However, they exhibit a slight decline in performance in the later epochs, especially in terms of f1-score and mAP@50.

All YOLOv5 variants demonstrate stability after epoch 50. Notably, the YOLOv5n variant not only exhibits better stability but also lower variability in all metrics beyond that epoch. The YOLOv5s variant also maintains strong performance throughout all epochs without significant performance reduction.

Regarding recall, which is the most relevant metric in our case, Table 8 presents the peak recall and corresponding epoch for each YOLOv5 variant in the context of wildfire and smoke detection. The results show a close range of recall rates between the models, with YOLOv5s leading slightly with a recall of 0.908. YOLOv5m and YOLOv5x exhibit strong performance as well, with recall values of 0.905 and 0.904 respectively. YOLOv5n and YOLOv5l both achieve a recall of 0.901, suggesting that all models effectively capture a high proportion of true positive instances and could be a suitable choice for our wildfire and smoke detection application.

The training epoch at which each model reaches its best recall varies. YOLOv5s peaks at epoch 226, suggesting that more extensive training can lead to slightly improved performance for this model. In contrast, YOLOv5m and YOLOv5x reach their best recall at relatively earlier epochs (126 and 131 respectively), indicating that they may optimize faster. YOLOv5n and YOLOv5l, with best recalls at epochs 175 and 132, also illustrate this range of optimization times across models. This suggests that monitoring model performance across epochs is crucial for optimal outcomes.

Finally, the average loss values for all the YOLOv5 variants addressed are displayed in Fig. 9.

Comparing the models, we observe a gradual decrease in average loss values as the model complexity increases. YOLOv5n exhibits higher loss values compared to other variants, while YOLOv5x demonstrates the lowest losses across all (box, obj, and cls). This suggests that the larger models, with more parameters and deeper architectures, are better equipped to learn and capture the intricate features relevant to smoke and wildfire detection.

Specifically, YOLOv5n shows an average training box loss of 0.0294 and a validation box loss of 0.0307. In contrast, YOLOv5x achieves an average training box loss of 0.0200 and a validation box loss of 0.0313, indicating a significant reduction in loss.

However, it is important to consider the phenomenon of overfitting as model complexity increases. Overfitting occurs when a model becomes overly specialized in capturing the training data, resulting in poor generalization on unseen data. Interestingly, YOLOv5l and YOLOv5x have slightly higher validation loss values for box, obj, and cls compared to the smaller variants. This suggests that the larger models may be more prone to overfitting, struggling to generalize effectively to new, unseen data.

To strike a balance between model complexity and generalization, YOLOv5m emerges as a viable option. It achieves relatively lower average loss values across all components compared to YOLOv5n and YOLOv5s, while also demonstrating competitive performance compared to YOLOv5l and YOLOv5x. YOLOv5m strikes a favorable trade-off between capturing important details and maintaining generalization capabilities, making it a suitable choice for smoke and wildfire detection tasks.

TABLE 9. Training time of each YOLOv6 variant.

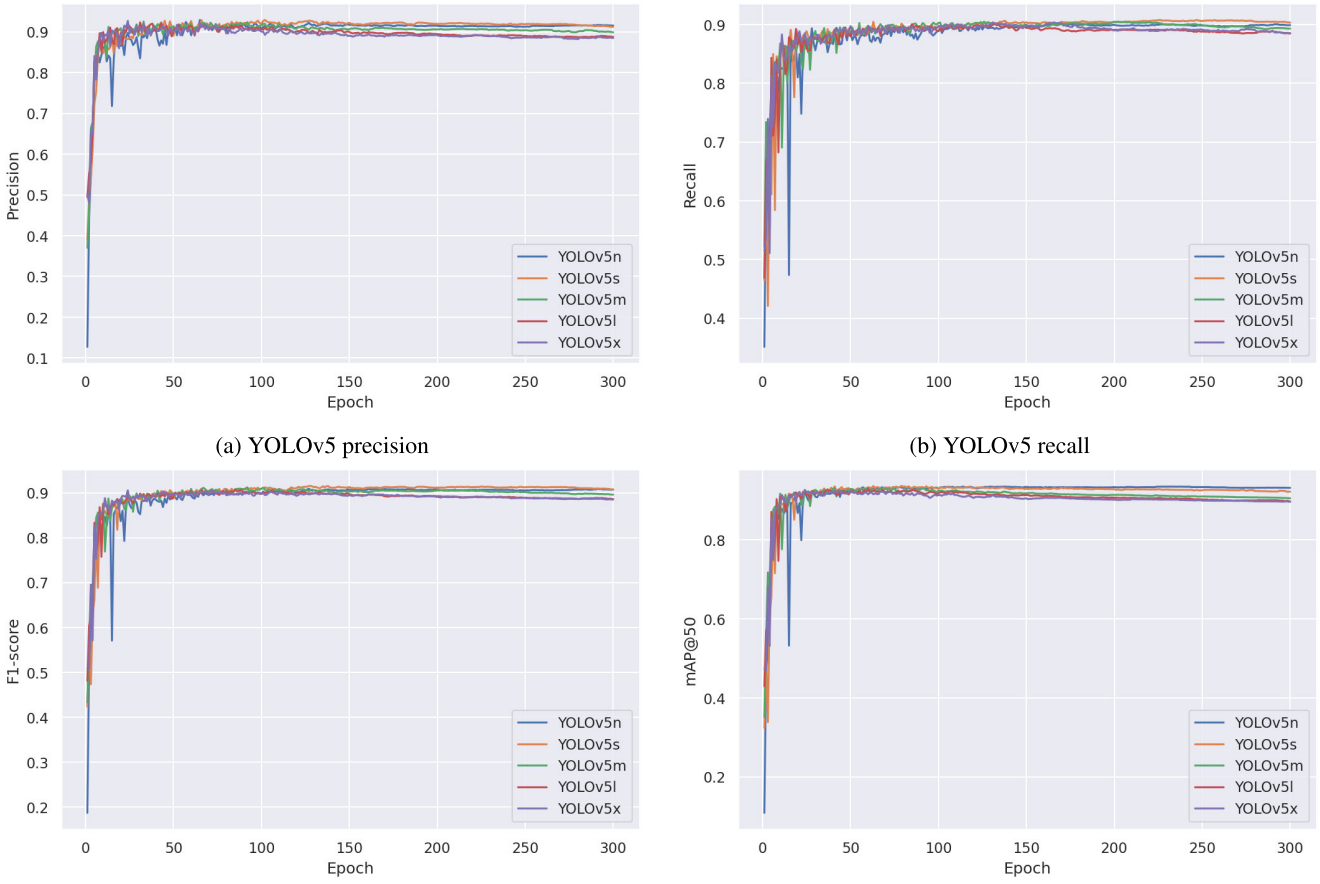
Model	Training time (hours)
YOLOv6n	**3.218**
YOLOv6s	3.301
YOLOv6m	4.230
YOLOv6l	4.795

B. YOLOv6

Table 9 presents the training times for each YOLOv6 variant. Notably, the differences in training times among the variants are relatively minor.

The YOLOv6n variant is the quickest, with a training time of just over 3.218 hours. Closely following is the YOLOv6s variant, which finishes its training process in approximately 3.301 hours, a minor increase from the YOLOv6n. A more significant increment is observed when moving to the YOLOv6m variant, which completes training in around 4.230 hours. This trend continues with the YOLOv6l variant, which has the longest training duration of the bunch at roughly 4.795 hours.

Despite these differences, all variants complete the training process within a reasonable time frame, making each a viable choice depending on the specific requirements of the task at hand.



(a) YOLOv5 precision

(b) YOLOv5 recall

(c) YOLOv5 F1-score

(d) YOLOv5 mAP@50

FIGURE 8. Evolution of all metrics for all YOLOv5 variants on validation.

Table 10 presents the performance metrics for each YOLOv6 variant on validation.

On close inspection, it's interesting to note the nuanced differences between each variant, despite the relatively minor variations in numerical terms. We see that YOLOv6l manages to attain the highest precision, F1-score, and mAP@50 among all the models. The highest precision score indicates that YOLOv6l model was capable of minimizing the number of false positives, a factor that could be crucial in scenarios where incorrect detections can have serious consequences.

On the other hand, YOLOv6m achieves the highest recall among the models, indicating its strength in identifying as many actual positives as possible. This could be particularly useful in scenarios where missing an actual positive could be detrimental. However, the trade-off for a higher recall is often a lower precision, as we can see with the YOLOv6m's slightly lower precision.

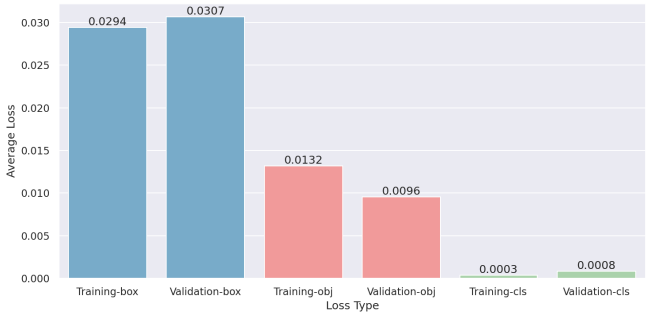
While the F1-score of the YOLOv6l is the highest, the YOLOv6m follows closely. It is interesting to note that YOLOv6l and YOLOv6m's values are almost identical, suggesting that these models have found a good trade-off between precision and recall under the given conditions.

TABLE 10. Performance metrics for each YOLOv6 variant on validation.

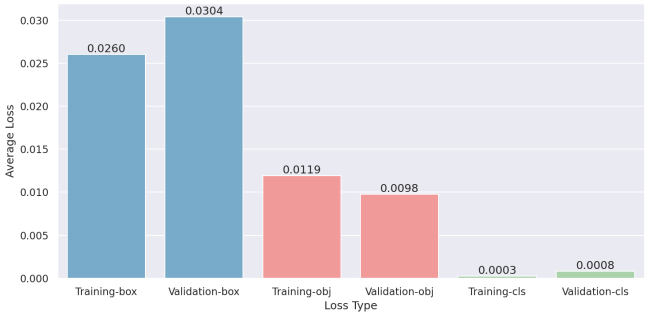
Model	Precision	Recall	F1-score	mAP@50
YOLOv6n	0.889	0.647	0.748	0.889
YOLOv6s	0.896	0.651	0.754	0.896
YOLOv6m	0.895	**0.654**	0.755	0.895
YOLOv6l	**0.900**	0.652	**0.756**	**0.900**

Finally, considering the mAP@50 values, all the models have achieved similar scores, which are quite high. This indicates the robustness of these models and their capability to reliably identify objects across a range of IoU thresholds. The fact that the mAP@50 scores are similar across the models, despite the variance in other metrics, demonstrates the adaptability of these models in maintaining consistent average precision across different use-cases.

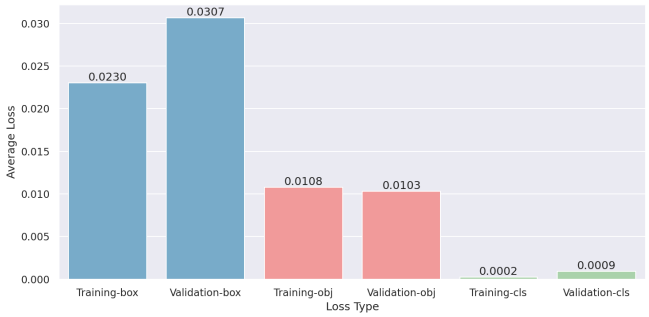
In summary, while all the YOLOv6 variants show high precision and mAP@50 scores, there exists a significant gap towards the recall. This insight suggests future iterations of these models could benefit from strategies focusing on improving recall values, such as modifying the loss function or adjusting the threshold for class predictions.



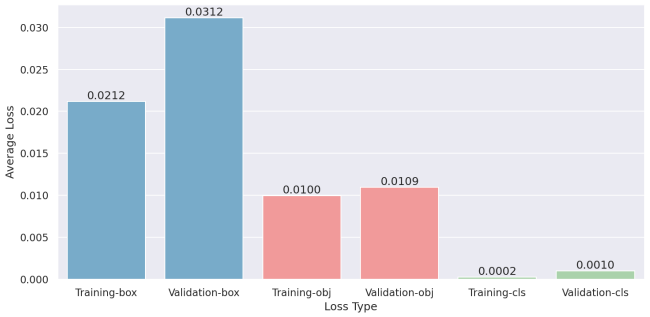
(a) YOLOv5n



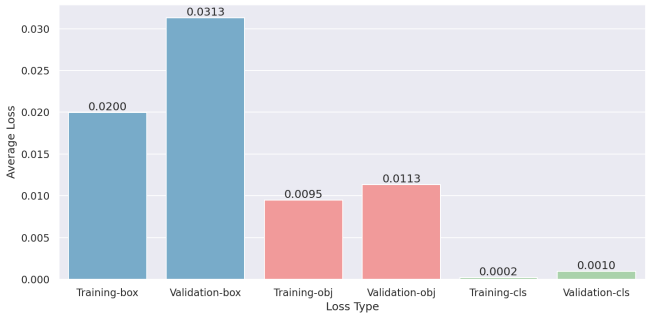
(b) YOLOv5s



(c) YOLOv5m



(d) YOLOv5l



(e) YOLOv5x

FIGURE 9. **Losses YOLOv5 variants.**

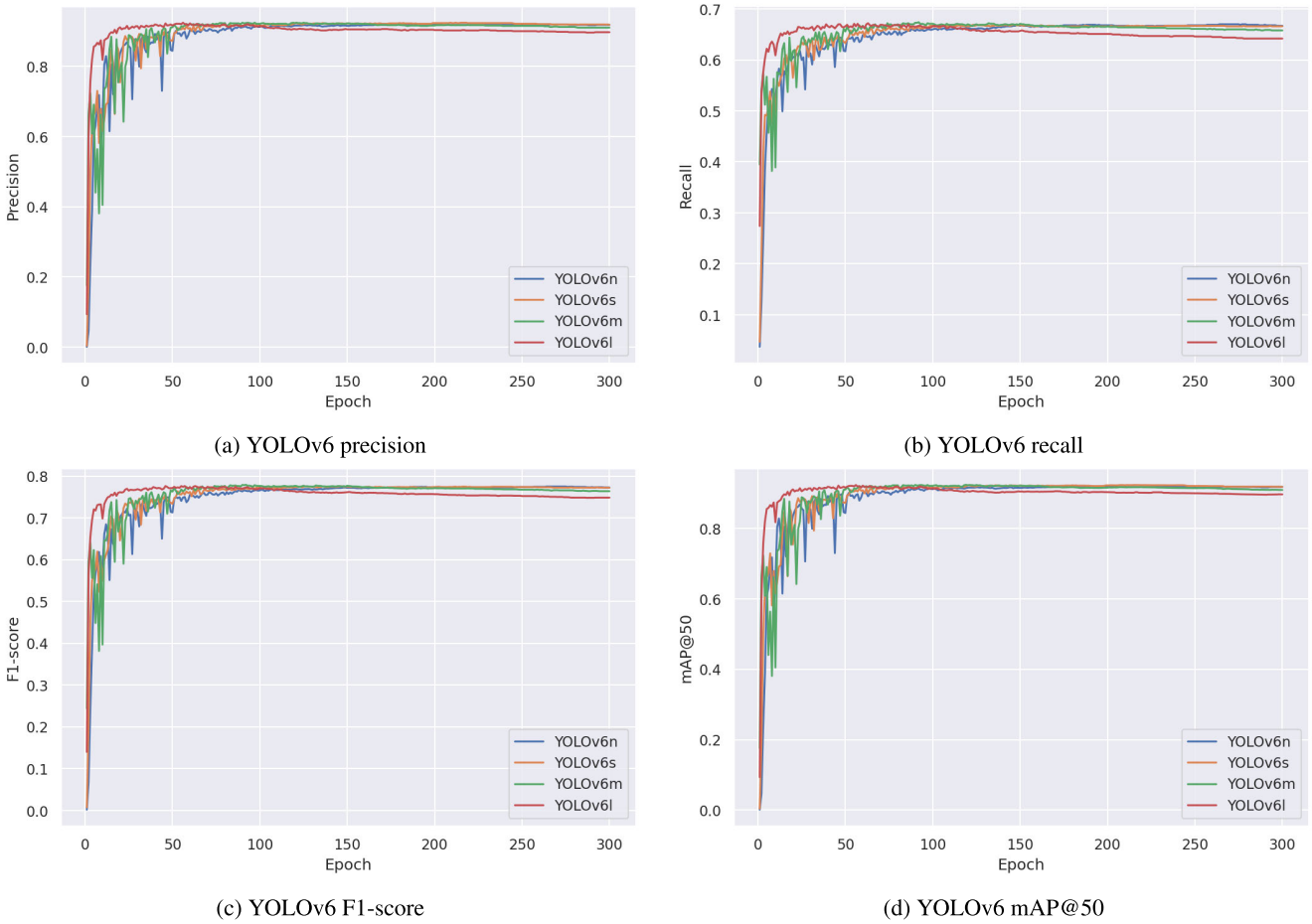
Analyzing the results of the metrics in depth, Fig. 10 illustrates the evolution of these metrics throughout the validation epochs. In this figure, it is evident that the YOLOv6l variant converges and stabilizes more rapidly than the others across all metrics. In contrast, YOLOv6m exhibits significant variability in the early epochs and takes longer to reach stability. The YOLOv6n and YOLOv6s variants show a consistent performance, displaying minimal variability in the early epochs and achieving stability relatively early.

However, despite being the most stable and fastest converging variant, YOLOv6l experiences a performance decline in the later epochs across all metrics. YOLOv6m also exhibits a performance drop in the later epochs, although less pronounced than YOLOv6l. In contrast, the YOLOv6n and YOLOv6s variants do not show a significant performance decline in the later epochs, maintaining consistent results.

Table 11 highlights the best recall obtained during the training epochs of each YOLOv6 variant. It can be observed that all models have reached their best recall relatively close to each other, ranging between 0.670 and 0.674, suggesting comparable performance in terms of detecting relevant objects.

The YOLOv6m variant managed to reach its highest recall of 0.674 in the 92nd epoch, which is relatively earlier than other models. This early peak performance could be an indicator of the efficiency of this particular model variant in learning and optimizing its recall. On the other hand, the YOLOv6n variant, despite achieving the lowest best recall of 0.670 among the group, did so at a later epoch. This could imply a slower convergence or require more training time to optimize its recall.

Meanwhile, YOLOv6s and YOLOv6l variants reached their best recall scores at the 123rd and 63rd epochs,



(a) YOLOv6 precision

(b) YOLOv6 recall

(c) YOLOv6 F1-score

(d) YOLOv6 mAP@50

FIGURE 10. Evolution of all metrics for all YOLOv6 variants on validation.

TABLE 11. Best recall and corresponding epoch for each YOLOv6 model on validation.

Model	Best recall	Epoch
YOLOv6n	0.670	264
YOLOv6s	0.669	123
YOLOv6m	**0.674**	92
YOLOv6l	0.672	63

respectively. The fact that the larger model reached its peak recall earlier than its smaller counterpart could point towards the benefits of its increased complexity and capacity in achieving high recall scores earlier in the training process.

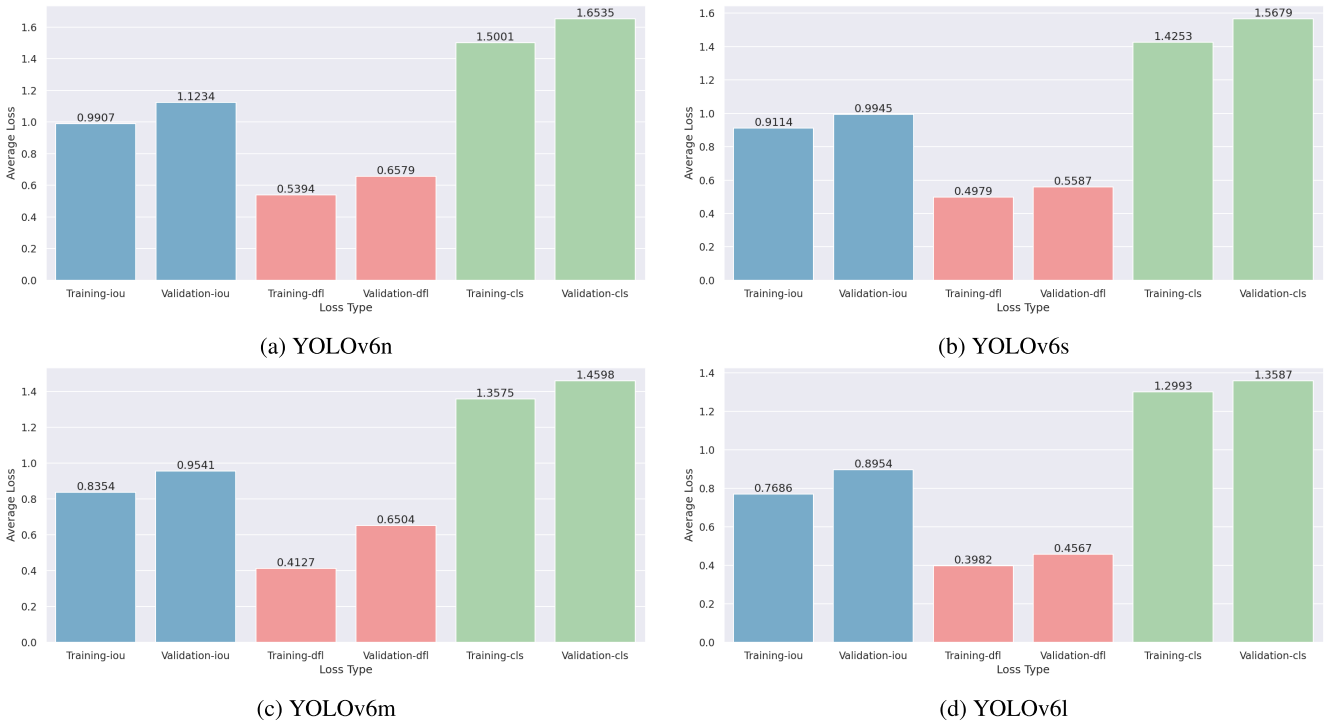
However, it's interesting to note that the overall difference in the best recalls across all four variants is minimal, pointing to a similar level of performance in terms of recall optimization.

Upon examining the average losses for each YOLOv6 variant, shown in Fig. 11, important patterns emerge. A key observation is that all three types of loss–IoU, dfl, and cls–tend to be higher in the validation set compared to the training set.

Looking at the variants individually, YOLOv6n reports the highest losses across all categories, indicating its relatively lower efficiency. Specifically, its IoU loss for validation is 1.1234, the highest among the four variants, and its cls loss for validation is also the highest at 1.6535, suggesting that it struggles more with class prediction on the validation set. On the other hand, its dfl loss in training is the second highest, indicating challenges with determining the correct bounding box dimensions in the training phase.

Conversely, YOLOv6l has the lowest losses across all categories, hinting at its superior performance over the other variants. Its IoU loss for validation, an indicator of how well the predicted bounding boxes align with the actual boxes, is the lowest at 0.8954. Additionally, it reports the lowest cls loss for validation, suggesting it is most effective at predicting the correct class.

YOLOv6s and YOLOv6m exhibit similar performances, with their losses hovering between those of YOLOv6n and YOLOv6l. Their IoU losses for validation are relatively close, 0.9945 and 0.9541 respectively, while their cls losses for validation are also similar, indicating comparable performances on class prediction.



(a) YOLOv6n

(b) YOLOv6s

(c) YOLOv6m

(d) YOLOv6l

FIGURE 11. Losses YOLOv6 variants.

TABLE 12. Training time of each YOLOv7 variant.

Model	Training time (hours)
YOLOv7	**5.969**
YOLOv7x	6.035

TABLE 13. Performance metrics for each YOLOv7 variant on validation.

Model	Precision	Recall	F1-score	mAP@50
YOLOv7	**0.887**	**0.869**	**0.878**	**0.903**
YOLOv7x	0.874	0.867	0.869	0.893

Loss missing: PRs no aceptados, feature (val loss) no disponible en los archivos originales

C. YOLOv7

In Table 12, the training time of each YOLOv7 variant is presented. It can be observed that YOLOv7 took approximately 5.969 hours to train while YOLOv7x took slightly longer, approximately 6.035 hours. This difference in training time directly relates to the enhanced complexity of the YOLOv7x architecture, as it contains more complex features and layers. However, the difference is relatively small, indicating that both models can be trained in a reasonable amount of time for applications in smoke and wildfire detection.

In Table 13, the performance of the YOLOv7 and YOLOv7x models is evaluated. The YOLOv7 model exhibits a slightly higher performance in each of these metrics, outperforming YOLOv7x. Specifically, YOLOv7 achieves a precision of 0.887 compared to 0.874 for YOLOv7x; therefore, YOLOv7's higher precision suggests that it may produce fewer false positives compared to YOLOv7x.

In terms of recall, the models show similar performance, with YOLOv7 reaching 0.869 and YOLOv7x slightly behind at 0.867. This similarity suggests that both models have comparable abilities to detect the majority of positive instances.

The F1-score is higher for YOLOv7 than for YOLOv7x, reinforcing the notion that YOLOv7 provides a more balanced performance in terms of identifying true positives and minimizing both false negatives and positives.

Finally, regarding mAP@50, YOLOv7 also surpasses YOLOv7x, scoring 0.903 compared to 0.893. This indicates that YOLOv7 not only detects more positive instances but also does so with a higher precision across various thresholds.

Overall, the results show that YOLOv7 outperforms YOLOv7x in the considered performance metrics, despite a slightly shorter training time. These results underline YOLOv7's effectiveness for smoke and wildfire detection tasks.

The evolution of the metrics throughout the validation epochs is visualized in Fig. 12. Upon analyzing the graph, it becomes evident that the YOLOv7 model exhibits a slower convergence rate compared to the YOLOv7x model. This discrepancy is particularly notable when examining the recall

TABLE 14. Best recall and corresponding epoch for each YOLOv7 model on validation.

Model	Best recall	Epoch
YOLOv7	0.900	68
YOLOv7x	**0.905**	262

metric, which displays more significant fluctuations during the initial epochs for the YOLOv7 model, as opposed to the more robust model.

In terms of stability, from around epoch 50 onwards, all models reach a state of stability; however, the YOLOv7x model exhibits a slightly higher degree of variability in its performance. It is worth noting that both models demonstrate stability, particularly in terms of F1-score and mAP@50, indicating their ability to maintain a consistent level of performance.

These findings shed light on the contrasting convergence characteristics and stability profiles of the YOLOv7 and YOLOv7x models. The slower convergence of the YOLOv7 model, as evidenced by its recall fluctuations in the early epochs, suggests that it may require more iterations to reach an optimal performance level. However, the YOLOv7 model stands out for its consistent overall performance, reflecting its ability to maintain a consistent level of accuracy, once it has reached its peak performance. In contrast, the YOLOv7x model exhibits a marginally higher degree of variability, suggesting that its performance may vary to a certain extent across different iterations.

Continuing with the analysis of the metrics, Table 14 presents the epoch at which each YOLOv7 model achieved the best recall score during validation. As mentioned above, recall is a significant metric for smoke and wildfire detection systems as it indicates the model's ability to correctly identify all instances of the positive class.

In this comparison, YOLOv7x achieves a higher recall of 0.905, slightly outperforming YOLOv7, which reaches a recall of 0.900. This suggests that YOLOv7x has a slightly stronger ability to correctly identify all smoke or wildfire instances, thus minimizing false negatives.

However, an important consideration is the difference in epochs when these peak recall scores were achieved. YOLOv7 hit its peak recall at the 68th epoch, whereas YOLOv7x required significantly more training and didn't reach its peak until the 262nd epoch. This discrepancy indicates that while YOLOv7x may achieve a higher recall, YOLOv7 reaches near-optimal recall significantly faster. Therefore, when considering computational efficiency alongside performance, YOLOv7 appears to provide a better trade-off, achieving comparable recall in a much shorter timeframe.

This insight could be crucial for applications where training time is a factor, as YOLOv7 offers a high recall rate while requiring less computational resources and time.

Fig. 13 compares the average losses during training and validation for both YOLOv7 and YOLOv7x models across box, obj, and cls losses. In both models, the validation loss is consistently higher than the training loss for all types of loss, suggesting that both models may be overfitting to the training data.

Looking at the box loss, both models show higher validation loss than training loss, with YOLOv7x exhibiting slightly higher box losses during both phases. This result suggests that both models could improve their bounding box prediction, with YOLOv7x having a slightly greater room for improvement.

Considering the obj loss, both models present a significant jump from training to validation loss, implying that they are overfitting on the training data. However, YOLOv7x has a lower validation obj loss, suggesting it may generalize better in detecting the presence of objects in the validation data, despite the overfitting.

Finally, for cls loss, the models again demonstrate a considerable increase in validation loss compared to training loss. YOLOv7x shows a lower validation cls loss than YOLOv7, indicating it might be better at classifying objects in the validation data.

Overall, the higher validation losses across all types for both models signal a case of overfitting. The models have learned the training data very well but are not generalizing effectively to unseen validation data. Even though YOLOv7x shows slightly higher training losses but lower validation losses, indicating better generalization, it's clear that both models could benefit from techniques to mitigate overfitting, such as more extensive data augmentation, dropout, or regularization methods.

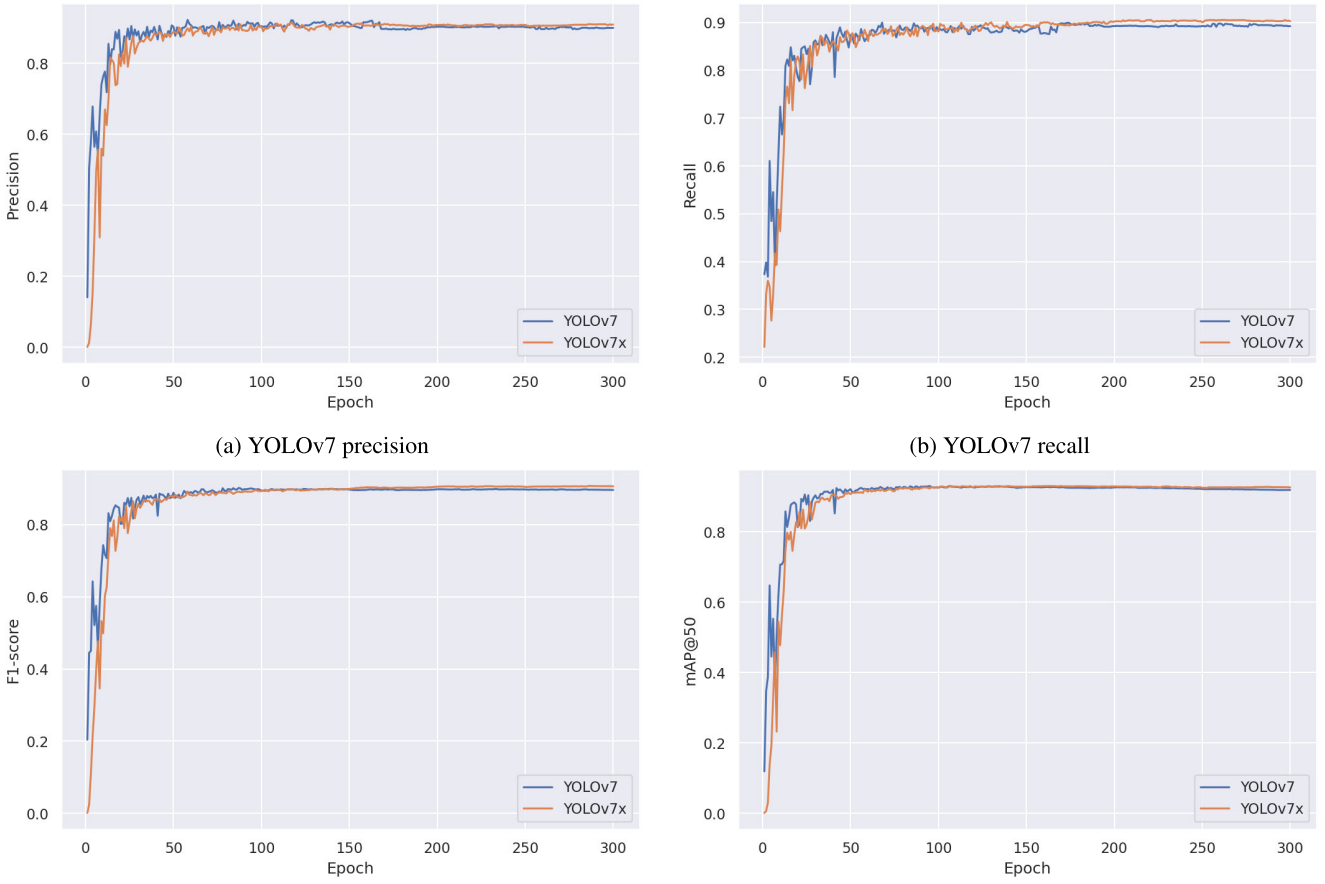
D. YOLOv8

Table 15 outlines the training time for each variant of the YOLOv8 model.YOLOv8n has the shortest training time, taking approximately 1.571 hours. This quick training time could make YOLOv8n an attractive choice for situations where rapid model deployment is needed, provided that its performance metrics do not lag significantly behind the other models.

YOLOv8s follows with a training time of 1.852 hours, which, while longer than YOLOv8n, is still relatively brief compared to the remaining models. The middle-ground model, YOLOv8m, takes significantly longer to train, with a training time of about 2.563 hours.

YOLOv8l's training time extends further to approximately 3.326 hours, and YOLOv8x, the model with the longest training time, clocks in at approximately 4.406 hours. These longer training times likely reflect more complex model architectures that may yield better performance metrics.

Table 16 presents a comparison of the performance metrics of YOLOv8 variants on the validation set. The YOLOv8n model achieves a precision of 0.886 and a recall of 0.855. These figures culminate in an F1-score of 0.870, and an mAP@50 of 0.905, demonstrating its overall effectiveness in object detection and localization.



(a) YOLOv7 precision

(b) YOLOv7 recall

(c) YOLOv7 F1-score

(d) YOLOv7 mAP@50

FIGURE 12. Evolution of all metrics for all YOLOv7 variants on validation.

(a) YOLOv7

(b) YOLOv7x

FIGURE 13. Losses YOLOv7 variants.

Shifting focus to the YOLOv8s model, there is a slight improvement in all metrics when compared to YOLOv8n, with precision at 0.888, recall at 0.869, F1-score at 0.878, and mAP@50 at 0.910. This suggests that despite its longer training time, YOLOv8s improves on the performance of YOLOv8n, likely due to increased model complexity.

The YOLOv8m model provides further incremental improvements in performance metrics, with a precision of 0.889, recall of 0.871, an F1-score of 0.879, and an mAP@50 of 0.909. As with the YOLOv8s model, these improvements, although marginal, suggest that the additional training time required for YOLOv8m could lead to slightly more accurate and reliable detections.

Interestingly, the performance metrics decrease slightly for the YOLOv8l model, with precision at 0.880, recall at 0.868, an F1-score of 0.874, and an mAP@50 of 0.901. Despite the

TABLE 15. Training time for each YOLOV8 model.

Model	Training time (hours)
YOLOv8n	**1.571**
YOLOv8s	1.852
YOLOv8m	2.563
YOLOv8l	3.326
YOLOv8x	4.406

longer training time, YOLOv8l does not show improvement over YOLOv8m, highlighting that longer training time does not always guarantee improved performance.

Finally, the YOLOv8x model, which had the longest training time, achieves a precision of 0.883, a recall of 0.875, an F1-score of 0.879, and an mAP@50 of 0.904. These metrics indicate that YOLOv8x performs comparably to YOLOv8m, but does not significantly outperform the other models despite its additional complexity and longer training time.

Overall, while there are marginal improvements in performance metrics across the YOLOv8 variants from YOLOv8n to YOLOv8m, these improvements are not strictly proportional to the increased complexity and training time.

TABLE 16. Performance metrics for each YOLOv8 variant on validation.

Model	Precision	Recall	F1-score	mAP@50
YOLOv8n	0.886	0.855	0.870	0.905
YOLOv8s	0.888	0.869	0.878	**0.910**
YOLOv8m	**0.889**	0.871	**0.879**	0.909
YOLOv8l	0.880	0.868	0.874	0.901
YOLOv8x	0.883	**0.875**	**0.879**	0.904

The progression of metrics throughout the validation epochs is depicted in Fig. 14.

All models exhibit considerable variability during the initial epochs, particularly in recall and precision. Metrics such as F1-score and mAp@50 demonstrate slightly less variability than the other metrics.

Finding stability proves to be a challenge for all models, especially for YOLOv8l and YOLOv8x. It is not until around epoch 100 that a semblance of stability is observed. However, this stability is primarily evident in terms of F1-score and mAP@50. Fluctuations in precision and recall are still apparent even in the later epochs. Furthermore, both precision and recall show a decline in performance towards the end of the training process.

In terms of convergence, the more robust models, YOLOv8l and YOLOv8x, demonstrate slower convergence across all metrics, with recall and precision showing the most noticeable delays in reaching stable performance. The lighter models YOLOv8n, YOLOv8s, and YOLOv8m show higher convergence speed, greater overall stability and show no alarming signs of decreasing performance across epochs.

The observed trends highlight the complex dynamics of convergence and stability in the evaluated models. The slower convergence and delayed stability observed in YOLOv8l and YOLOv8x suggest that these models require more training iterations to achieve optimal performance. Despite the fluctuations, the models eventually demonstrate a level of stability, particularly in terms of F1-score and mAP@50.

Continuing with the analysis, Table 17 presents the highest achieved recall and the corresponding epoch for each variant of the YOLOv8 model during validation. The lightest model, YOLOv8n, reaches a competitive recall of 0.889 as early as epoch 108. Given its relative simplicity, it demonstrates quick learning and substantial detection capabilities. This swift convergence suggests that it might be a viable option for scenarios where computational resources and time are limited, without significant compromise in recall.

The YOLOv8s model, despite having more complexity than YOLOv8n, manages to achieve a slightly higher recall of 0.896 at epoch 128. This indicates that the model's additional complexity does contribute to better recall, but it also requires more epochs to learn, even though the increase in learning time is relatively modest.

Looking at the YOLOv8m model, despite achieving a recall of 0.894 at a much later epoch 270, it doesn't deliver a substantial improvement over YOLOv8s or even YOLOv8n. This may lead us to question the efficiency of additional complexity in this variant, as it doesn't correspond with better performance despite extended training.

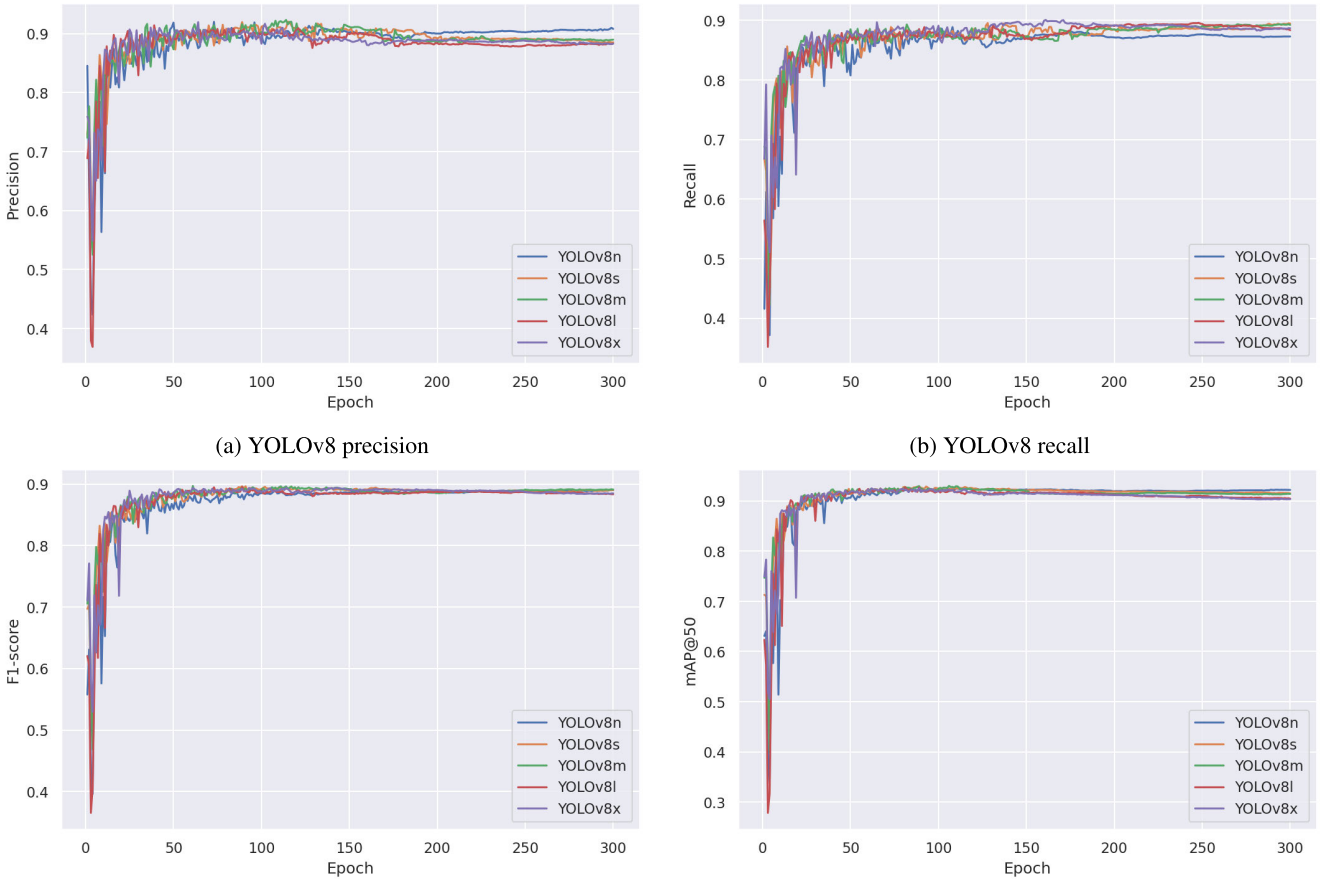
Interestingly, the YOLOv8l model achieves a recall of 0.896 at epoch 248, comparable to YOLOv8s, but the peak is reached at a later epoch. This suggests that even though YOLOv8l has more complexity than YOLOv8s, this does not necessarily translate into improved performance or faster learning.

Lastly, YOLOv8x, the most complex model, achieves the highest recall of 0.901 at epoch 161, outperforming all other variants in recall performance. Despite its increased complexity, it reaches its peak performance faster than both YOLOv8m and YOLOv8l. This indicates that YOLOv8x might strike the best balance between complexity and performance among these models, providing high detection accuracy without significantly increasing the training time.

These results underline that there is no direct relationship between model complexity and recall performance. Indeed, the highest recall is achieved by the most complex model (YOLOv8x), but not all increases in complexity lead to better performance, as seen in YOLOv8m and YOLOv8l.

Turning to the loss analysis, Fig. 15 shows the average losses during training and validation for all YOLOv8 variants.

All models show increased losses in validation compared to training across all three types: box, dfl, and cls. This gap suggests some degree of overfitting, where models learn the training data well but struggle to generalize these findings to unseen data. However, the extent of this divergence differs between the models and loss types, hinting at unique model characteristics.



(a) YOLOv8 precision

(b) YOLOv8 recall

(c) YOLOv8 F1-score

(d) YOLOv8 mAP@50

FIGURE 14. Evolution of all metrics for all YOLOv8 variants on validation.

TABLE 17. Best recall and corresponding epoch for each YOLOv8 model on validation.

Model	Best recall	Epoch
YOLOv8n	0.889	108
YOLOv8s	0.896	128
YOLOv8m	0.894	270
YOLOv8l	0.896	248
YOLOv8x	**0.901**	161

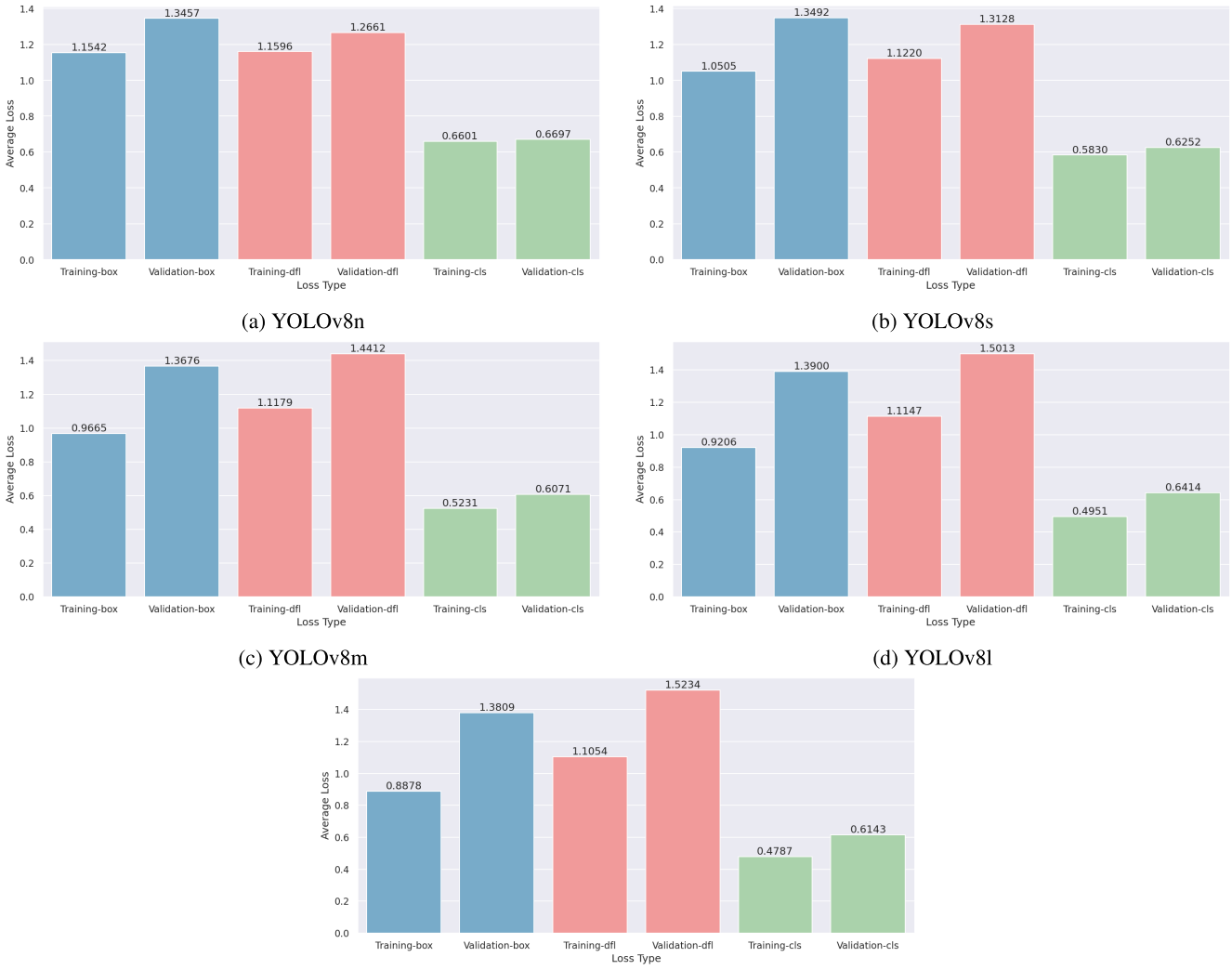
Starting with the box loss, we see a consistent pattern across all variants: as the model complexity increases, the training box loss gradually decreases. This decrease suggests that complex models can better learn to predict bounding boxes. However, in contrast, the validation box loss doesn't show a similar consistent decrease. While YOLOv8x, the most complex model, indeed has a lower validation box loss compared to YOLOv8n, it's not the lowest among all models. This discrepancy signals that the additional complexity doesn't always lead to better generalization in bounding box prediction.

The dfl shows a different trend. The training dfl is relatively consistent across all variants, showing minor reductions with increasing complexity. This could mean that all models are roughly equally capable of learning this aspect. Interestingly, the validation dfl generally increases with complexity. This could indicate overfitting, as complex models might be capturing noise or irrelevant patterns in the training set that don't generalize well.

The cls loss follows a somewhat predictable trend similar to the 'box' loss. More complex models perform better during training, reflected by the decreasing trend in training cls loss. However, the validation cls loss again doesn't display a clear pattern. While the most complex model, YOLOv8x, shows a lower validation cls loss than YOLOv8n, it doesn't outperform all other models. This suggests that while complexity might help with class prediction in training, it doesn't necessarily lead to better generalization.

E. YOLO-NAS

Examining Table 18, we observe the impact of the model's size on the training time for the YOLO-NAS variants. As anticipated, there is a direct relationship between the size of the model and the time taken for training.



(a) YOLOv8n

(b) YOLOv8s

(c) YOLOv8m

(d) YOLOv8l

(e) YOLOv8x

FIGURE 15. Losses YOLOv8 variants.

The smallest model, YOLO-NASs, requires the least training time, clocking in at approximately 2.427 hours. As we progress to the medium-sized model, YOLO-NASm, the training time increases to 3.460 hours. This is an increase of about 42.5%, which is a substantial rise considering the fact that we're dealing with a middle-sized model.

The trend continues as we transition to the largest model, YOLO-NASl. The training time jumps to 4.375 hours, which is about 26.9% more than the medium model and approximately 80.2% longer than the smallest model. This significant increase in training time underscores the computational demands associated with training larger models.

The performance metrics presented in Table 19 highlight interesting facets of the behavior of the different YOLO-NAS variants.

First, it is worth mentioning that the Recall scores for all models are exceptionally high, above 0.96 in all cases, which means that these models are very good at detecting the majority of actual positive cases of smoke and wildfires. This could be crucial in scenarios where the priority is to detect as many positive cases as possible, even at the risk of triggering some false alarms.

TABLE 18. Training time of each YOLO-NAS variant.

Model	Training time (hours)
YOLO-NASs	**2.427**
YOLO-NASm	3.460
YOLO-NASl	4.375

The Precision scores, however, tell a different story. They are markedly lower for all models, especially when compared to the high Recall scores. YOLO-NASl, the largest model, offers the best Precision at 0.107, followed by YOLO-NASm at 0.100, and finally, YOLO-NASs at 0.079. These low

Precision scores suggest that while the models are good at capturing positive cases, they also have a high rate of false positives.

The F1-score, grows with the model size, reflecting that larger models achieve a better balance between these two measures, albeit still very skewed towards Recall due to the lower Precision scores. Regarding the mAP@50 scores, all models perform similarly, with scores above 0.86. YOLO-NASl marginally outperforms the other models, reinforcing the notion that larger models deliver a slight edge in overall performance.

In summary, while all the YOLO-NAS models exhibit strong ability in capturing positive cases, they struggle with precision, indicating a high level of false positives. The models' complexity seems to positively influence precision and F1-score, albeit marginally, suggesting potential benefits of using larger models if computational resources and training time permit. However, the precision-recall trade-off must be carefully considered based on the specific requirements of the detection task.

TABLE 19. Performance metrics for each YOLO-NAS variant on validation.

Model	Precision	Recall	F1-score	mAP@50
YOLO-NASs	0.079	0.970	0.145	0.862
YOLO-NASm	0.100	0.968	0.180	0.872
YOLO-NASl	**0.107**	**0.971**	**0.192**	**0.873**

Moving on to a more detailed analysis of the metrics, Fig. 16 illustrates the evolution of these metrics during the validation epochs.

The figure clearly shows a distinct and noteworthy behavior between precision and recall across all YOLO-NAS variants. While recall converges rapidly and maintains a consistently high performance, precision, on the contrary, remains low throughout all epochs and exhibits considerable variability even in the later stages. Although precision shows improvement over the epochs, it is challenging to define the epoch at which stability is achieved, in contrast to recall, where stability becomes evident around epoch 10.

In the context of wildfire and smoke detection, a high recall indicates that the models are effective at identifying a significant proportion of true positive instances, which is crucial for ensuring comprehensive coverage and minimizing the risk of missing actual instances of wildfires and smoke. However, the observed low precision signifies that the models may also generate a considerable number of false positive detections. These false positives can lead to unnecessary alerts or additional resource allocation for verification, which can be a practical challenge in real-world scenarios with limited resources and the need for prompt and accurate response.

Regarding the F1-score, given the peculiar behavior of precision and recall, it displays a similar pattern to precision. This is because the F1-score is influenced by both precision and recall. With a high and constant recall while precision fluctuates, the F1-score primarily reflects the variability in precision.

TABLE 20. Best recall and corresponding epoch for each YOLO-NAS model on validation.

Model	Best recall	Epoch
YOLO-NASs	0.983	157
YOLO-NASm	0.980	104
YOLO-NASl	**0.984**	75

Shifting focus to mAP@50, we observe a similar trend to recall. All YOLO-NAS variants converge rapidly and maintain stability throughout the epochs. Although there is some minor variability, it is slightly more pronounced in the YOLO-NASs and YOLO-NASl variants. Similar to recall, there is no significant performance decline throughout the epochs, indicating a consistent performance across all variants.

These findings, within the context of wildfire and smoke detection, provide valuable insights into the performance dynamics of the YOLO-NAS models. The contrasting convergence and stability patterns between precision and recall highlight the trade-off between these metrics and the challenges inherent to accurately detecting wildfires and smoke instances. Understanding the nuances and characteristics of these metrics in this specific context is crucial for further research and improvement in developing more effective wildfire and smoke detection systems.

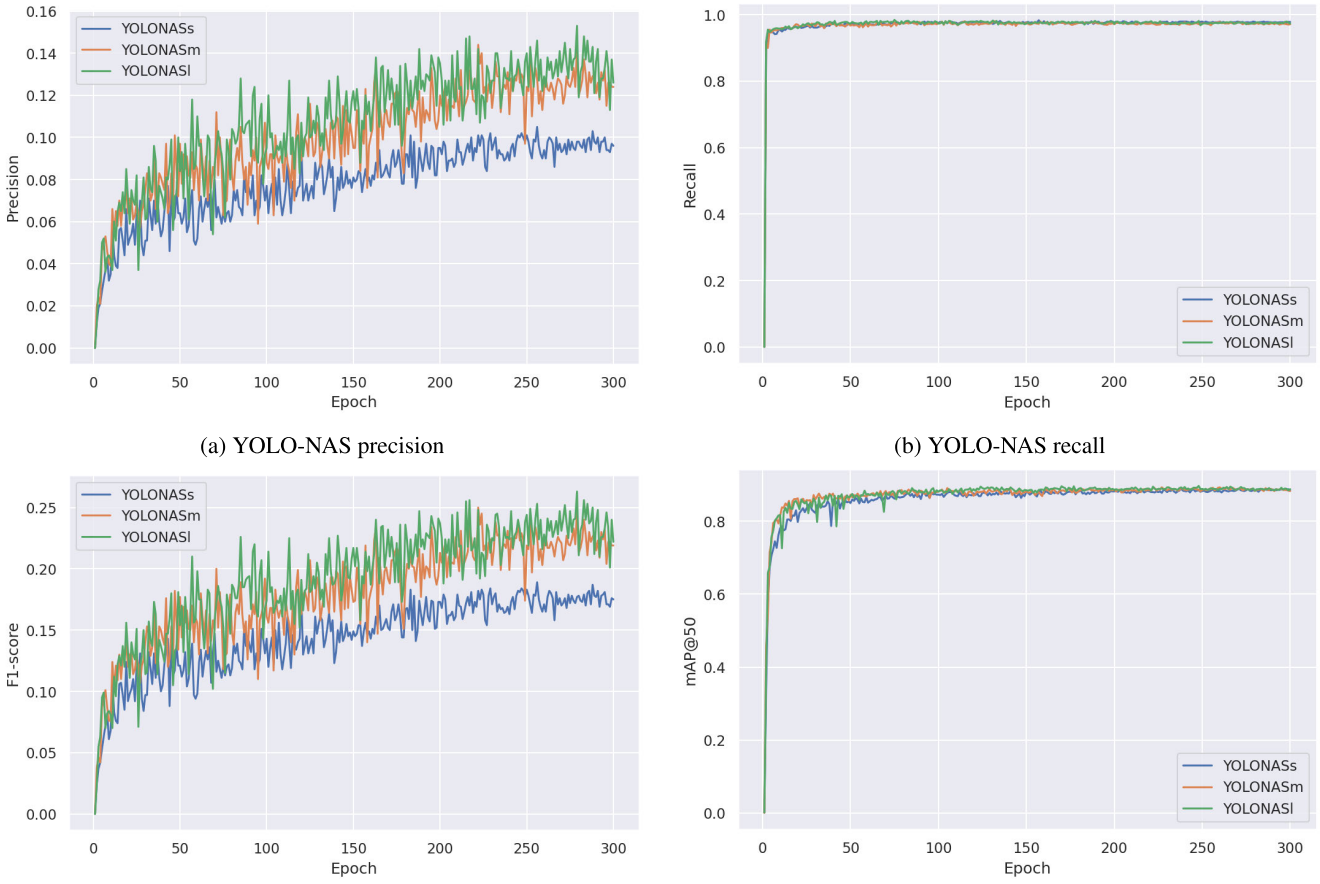
Analyzing Table 20, we can see a continuation of the trend observed previously in the high recall values achieved by all YOLO-NAS models. However, this table presents a more nuanced picture by considering the epoch at which each model achieved its best recall.

The largest model, YOLO-NASl, achieved the highest recall of 0.984, slightly outperforming the other two models. Intriguingly, it achieved this score at the 75th epoch, which is significantly earlier than the other models. This suggests a faster convergence towards an optimal model, likely due to its increased complexity and capacity to model the problem. This could be an important factor when computational resources or time are limited.

YOLO-NASm reached its best recall of 0.980 at the 104th epoch, while YOLO-NASs, the smallest model, peaked at a recall of 0.983 at the 157th epoch. Despite the smaller model size, YOLO-NASs surprisingly achieved a higher recall than the medium model, albeit at a later epoch.

Overall, the YOLO-NAS variants demonstrate an excellent ability to capture the majority of actual positive cases, with differences in the speed of convergence seemingly linked to model size. As such, the selection between these models would be influenced by the specific demands of the application, the resources available, and the time allowable for model training.

Moving on with the analysis, Fig. 17 shows the average losses for each YOLO-NAS variant.



(a) YOLO-NAS precision

(b) YOLO-NAS recall

(c) YOLO-NAS F1-score

(d) YOLO-NAS mAP@50

FIGURE 16. Metrics evolution YOLO-NAS variant.

In terms of dfl, the YOLO-NASm variant achieves the lowest average training and validation losses, at 0.9126 and 1.0038, respectively. This is closely followed by the YOLO-NASl model with respective losses of 0.9317 and 1.0189, and the YOLO-NASs model with losses of 0.9653 and 1.0357. Despite the slight differences, all models demonstrate reasonable performance, with losses remaining relatively low and suggesting good effectiveness in handling the imbalanced data distribution.

Turning to IoU loss, all models perform comparably with minor variances. YOLO-NASl reports the smallest average training loss of 0.1750, while YOLO-NASs records the highest with 0.1881. During validation, the models' performance appears relatively close, with all maintaining their losses below 0.205, indicating a good compromise between true positive and false positive rates.

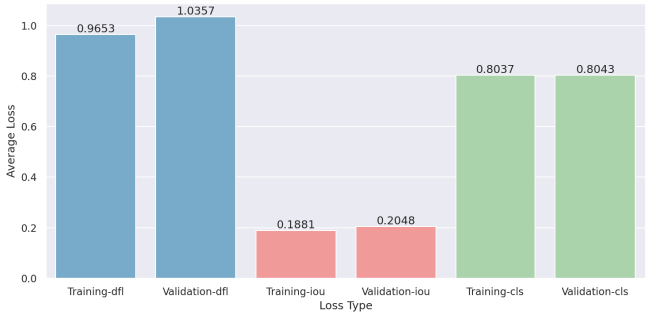
The training and validation cls losses, also reveal a similar pattern. YOLO-NASm and YOLO-NASl have almost identical losses, while YOLO-NASs shows slightly higher but still competitive losses. These figures indicate a strong performance in terms of correct object classification across all models.

Overall, these results suggest that all YOLO-NAS variants perform comparably across the evaluated metrics. YOLO-NASm seems to edge out the others in terms of dfl and cls loss, which could make it more appealing for applications dealing with imbalanced datasets and requiring precise class discrimination. However, the difference is small, and the choice between models would likely depend on the specifics of the application, and a balance between performance and computational resources.

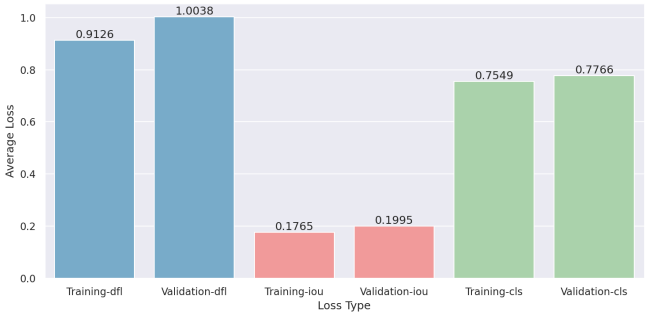
F. COMPARISON BETWEEN BEST MODELS

The YOLOv5 architecture shows a range of performances when evaluated on our wildfire and smoke detection test set, as seen in Table 21. Among the variants, the YOLOv5s model exhibits the highest recall of 0.893, a crucial metric for this application, as it signifies the model's proficiency in correctly identifying the majority of positive cases - essential for reducing missed detections in wildfire scenarios.

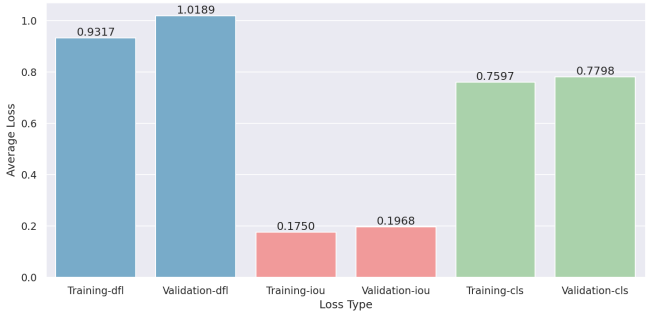
It also secures the highest precision of 0.895, outperforming the other variants, which indicates its strong ability to avoid false alarms. Consequently, these two metrics



(a) YOLO-NASs



(b) YOLO-NASm



(c) YOLO-NASl

FIGURE 17. Losses YOLO-NAS variants.

contribute to YOLOv5s achieving the best F1-score of 0.894, demonstrating a balance between precision and recall.

Interestingly, while YOLOv5s leads in recall, precision, and F1-score, the YOLOv5n model offers the best mAP@50 performance, achieving 0.916. This suggests that YOLOv5n manages to maintain a good balance between precision and recall across various object detection thresholds. Moreover, it offers the lowest inference time of 1.3 ms, indicating potential advantages for real-time applications.

As for the YOLOv5m and YOLOv5l models, their performance falls between the extremes of YOLOv5s and YOLOv5x. YOLOv5l, for example, exhibits a marginally higher recall than YOLOv5m, but shares the same precision. Yet, YOLOv5l pays the price in terms of inference time, being approximately 27% slower than YOLOv5m. This reveals a trade-off scenario between detection capability and processing speed.

Conversely, despite its size, the YOLOv5x model lags in all metrics, possibly due to overfitting during the training process. Its performance should be further scrutinized before deployment.

In the context of our wildfire detection application, where recall is of utmost importance, the YOLOv5s model, boasting the highest recall, is deemed the 'best global model' from the YOLOv5 architecture variants.

Examining the performance of the YOLOv6 variants, shown in Table 22, on our wildfire and smoke detection test set reveals a similar pattern of diverse performance. Among the variants, the YOLOv6m model secures the highest recall

TABLE 21. Metrics of the best models for each variant of the YOLOv5 architecture on testing.

Model	Precision	Recall	F1-score	mAP@50	Inference time (ms)
YOLOv5n	0.893	0.884	0.888	**0.916**	**1.3**
YOLOv5s	**0.895**	**0.893**	**0.894**	0.905	1.8
YOLOv5m	0.883	0.883	0.883	0.895	2.6
YOLOv5l	0.883	0.886	0.885	0.905	3.3
YOLOv5x	0.874	0.875	0.875	0.875	5.6

of 0.890. Simultaneously, YOLOv6n claims the highest precision of 0.919, suggesting a robust ability to avert false alarms. This contributes to the model's F1-score of 0.895, signifying a strong balance between precision and recall.

Additionally, YOLOv6n outperforms the other variants in mAP@50 by achieving 0.915, indicating its adeptness at maintaining high precision and recall across different object detection thresholds. YOLOv6n also holds the fastest inference time of 0.54 ms, suggesting it as a potential candidate for real-time applications. However, it should be noted that it does not lead in recall, the most important metric for our task.

YOLOv6s and YOLOv6l provide middling performances in terms of recall and precision. However, it's interesting to note that YOLOv6l, despite its slower inference time, outperforms YOLOv6s in both recall and precision, suggesting that some trade-off between detection ability and processing speed has been made.

TABLE 22. Metrics of the best models for each variant of the YOLOv6 architecture on testing.

Model	Precision	Recall	F1-score	mAP@50	Inference time (ms)
YOLOv6n	**0.919**	0.872	0.895	0.915	**0.54**
YOLOv6s	0.909	0.869	0.889	0.906	0.99
YOLOv6m	0.905	**0.890**	**0.897**	**0.916**	1.85
YOLOv6l	0.889	0.880	0.884	0.907	2.75

Considering the task at hand, where high recall is paramount, the YOLOv6m model, with the highest recall, is considered the 'best global model' from the YOLOv6 architecture variants.

Upon evaluating the YOLOv7 architecture and its variant YOLOv7x on the wildfire and smoke detection test set, we find both models presenting commendable performances, as shown in Table 23. The YOLOv7 model exhibits a precision of 0.904, the highest within this architecture. This suggests that, among its detections, a high proportion of true positives ensures a low false alarm rate, which is particularly crucial when dealing with wildfire detection systems.

However, the YOLOv7x variant surpasses the base model in terms of recall, reaching a value of 0.902. This indicates a superior ability of YOLOv7x to correctly identify wildfires and smoke instances within the images.

The two models are quite balanced in terms of the F1-score, both achieving 0.887, indicating a balanced trade-off between precision and recall. Also, their performance in mAP@50 is quite similar, with YOLOv7x slightly edging out at 0.911 compared to YOLOv7's 0.910, which again suggests a slightly superior overall object detection capability.

The inference times of YOLOv7 and YOLOv7x are 2.7 ms and 2.8 ms respectively, representing only a minor difference. Given the substantial improvement in recall and the negligible increase in inference time, the YOLOv7x variant can be considered the 'best global model' among the YOLOv7 architecture variants for our specific task of wildfire and smoke detection.

TABLE 23. Metrics of the best models for each variant of the YOLOv7 architecture on testing.

Model	Precision	Recall	F1-score	mAP@50	Inference time (ms)
YOLOv7	**0.904**	0.870	**0.887**	0.910	**2.7**
YOLOv7x	0.873	**0.902**	**0.887**	**0.911**	2.8

The YOLOv8 architecture variants demonstrate varied performance when evaluated on the test set, as seen in Table 17. YOLOv8s, despite its smaller size compared to other variants, achieves the highest precision of 0.905, meaning that its predictions are largely accurate and it effectively minimizes false positives. This is crucial in the context of wildfire detection, where a false positive could potentially lead to unnecessary panic or resource allocation.

On the other hand, YOLOv8n stands out with the highest recall of 0.889, which indicates that it successfully identifies a significant majority of actual instances of wildfires and smoke in the test set. As previously discussed, high recall is crucial in our application, as the failure to detect a genuine fire could lead to disastrous outcomes.

Despite these strengths, no single variant clearly dominates across all the metrics. For example, while YOLOv8s has the best precision and F1-score, YOLOv8n excels in terms of recall and has the fastest inference time, which is important for real-time detection systems. However, YOLOv8s achieves the highest mAP@50 score, implying a better trade-off between precision and recall across different thresholds.

When observing the inference times, it is evident that the size and complexity of the model have a direct impact on the computation time. YOLOv8n has the shortest inference time of 0.8 ms, while YOLOv8x, the largest model, requires 4.4 ms.

Considering the significance of recall in our context and the necessity for fast inferencing, the YOLOv8n variant, with its superior recall and the fastest inference time, is selected as the 'best global model' within the YOLOv8 architecture for wildfire and smoke detection. It provides a satisfactory balance between detection accuracy and computational efficiency, making it most suited for this particular application.

TABLE 24. Metrics of the best models for each variant of the YOLOv8 architecture on testing.

Model	Precision	Recall	F1-score	mAP@50	Inference time (ms)
YOLOv8n	0.866	**0.889**	0.877	0.909	**0.8**
YOLOv8s	**0.905**	0.869	**0.887**	**0.913**	1.2
YOLOv8m	0.884	0.863	0.873	0.897	2.0
YOLOv8l	0.870	0.883	0.876	0.893	2.9
YOLOv8x	0.882	0.870	0.876	0.890	4.4

Moving forward, the performance metrics of the YOLO-NAS architecture variants present a fascinating trade-off between precision and recall, as seen in Table 20. Same as seen in training, these models have extraordinarily high recall scores, nearly reaching the maximum possible value of 1 for each variant. This indicates that the models are exceptionally good at detecting actual wildfire and smoke events in the test set.

However, the precision scores of these models are remarkably low, with all three variants scoring less than 0.1. This suggests that these models produce a large number of false positives, identifying many non-fire events as fires. As mentioned above, these false alarms can lead to misallocation of resources, creating an inefficient system.

These trade-offs significantly impact the F1-score, which strives to balance precision and recall. Despite the high recall, the F1-scores for all variants are very low, the highest being only 0.151 for YOLO-NASs, due to the poor precision.

Furthermore, the Mean Average Precision (mAP@50) scores hover around 0.85, reflecting the models' balance

(a) Ground-truth (b) YOLOv5s (c) YOLOv6m

(d) YOLOv7x (e) YOLOv8n (f) YOLO-NASm

FIGURE 18. First inference test with the best global models. Image taken from the test dataset.

between precision and recall at different thresholds. The inference times range from 1.06 ms for the YOLO-NASs to 1.52 ms for YOLO-NASl, a negligible difference for real-time detection systems.

In summary, the YOLO-NAS architecture's high recall might be desirable in scenarios where missing a positive instance can have serious consequences, such as wildfire detection. However, the considerably low precision suggests a high rate of false alarms, which could be problematic. Consequently, employing these models would require careful consideration of the potential impact of false positives in the specific use-case. However, taking into account the recall, we selected the YOLO-NASm model as the best overall model in this case.

TABLE 25. Metrics of the best models for each variant of the YOLO-NAS architecture on testing.

Model	Precision	Recall	F1-score	mAP@50	Inference time (ms)
YOLO-NASs	**0.082**	0.971	**0.151**	0.859	**1.06**
YOLO-NASm	0.062	**0.974**	0.115	**0.861**	1.32
YOLO-NASl	0.079	0.971	0.146	0.853	1.52

Now, we use each of the 'best global models' (YOLOv5s, YOLOv6m, YOLOv7x, YOLOv8n, YOLO-NASm) to make inferences and produce some graphical results, which are shown in Figs. 18, 19 and 20.

First, regarding Fig. 18, we observe a scene of a distant smoke column. All models correctly recognize the object, and the bounding box is accurately placed. Additionally, it can be observed that the YOLOv5s, YOLOv6m, and YOLOv7x architectures exhibit slightly higher confidence, with a value of 0.87, compared to the YOLOv8n and YOLO-NASm architectures, which obtained a confidence of 0.82. This difference in confidence levels may suggest that YOLOv5s, YOLOv6m, and YOLOv7x architectures have higher certainty in their detections compared to the other two. However, the difference is small, and at least in this specific test, all architectures demonstrate a comparable performance in smoke recognition.

Moving on to Figure 19, we observe a scene that contains both flames and smoke. In the ground-truth (a), we can see two instances of smoke and one instance of fire. Interestingly, the YOLOv6m model stands out for correctly recognizing all instances in the ground-truth and even identifying fire in other areas of the image. This is particularly remarkable since the YOLOv6m architecture variants showed low recall levels during training and validation. However, they seem to perform well during testing.

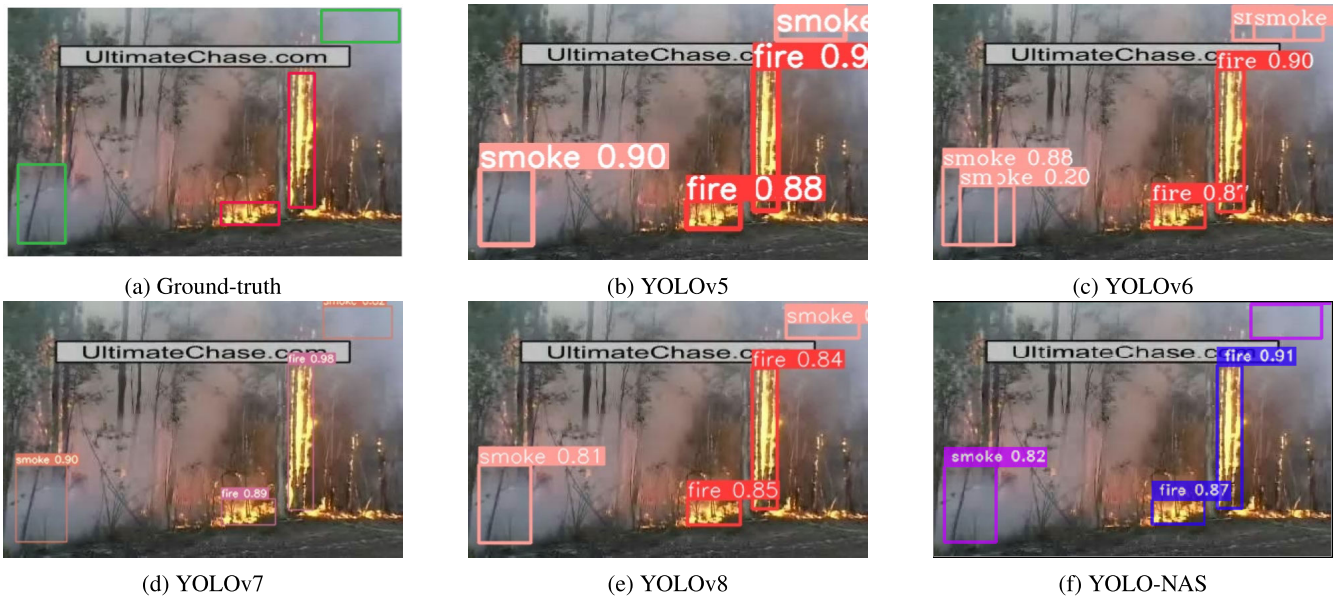
Next, the YOLOv5s and YOLOv8n architectures also adequately recognize fire and smoke. However, the confidence levels of the bounding boxes are lower compared to YOLOv6m. On the other hand, YOLOv7x fails to detect one instance of smoke, and the confidence levels are much lower than the aforementioned architectures. Furthermore, YOLO-NASm fails to detect the fire instance and only identifies the smoke. This is surprising because all YOLO-NASm variants showed the best numbers in terms of recall.

Moving on to Fig. 20, we observe a scene with four instances, two of fire and two of smoke. Here, the most evident aspect is that all models correctly detect the instances of interest. YOLOv6m stands out for detecting more instances than those shown in the ground-truth. Although this model does not recognize instances in incorrect locations, it appears that some instances overlap. In other words, in certain contexts, this model may not accurately determine the exact

(a) Ground-truth (b) YOLOv5s (c) YOLOv6m

(d) YOLOv7x (e) YOLOv8n (f) YOLO-NASm

FIGURE 19. Second inference test with the best global models. Image taken from the test dataset.



(a) Ground-truth (b) YOLOv5 (c) YOLOv6

(d) YOLOv7 (e) YOLOv8 (f) YOLO-NAS

FIGURE 20. Third inference test with the best global models. Image taken from the test dataset.

boundaries of the object. The other models precisely recognize the four instances in the ground-truth. The difference among them lies in the confidence level of the bounding boxes. YOLOv5s, YOLOv6m, and YOLOv7x exhibit a slightly higher confidence level than YOLOv8n and YOLO-NASm.

Interestingly, despite this scene containing fire and smoke similar to Fig. 18, here the models correctly recognized all instances in the ground-truth, something that did not occur in the first example. This could be attributed to the type of scene, as this is a closer shot, while Fig. 18 represents a more distant or panoramic scene. This difference may be due to a lack of diversity in the training data, specifically in panoramic scenes, which could limit the models' ability to accurately detect instances in that context.

Now, the previous examples consisted of images taken from the test set that share certain similarities with the training and validation data. Therefore, we conducted additional tests using freely available images obtained from the internet, which are shown in Fig. 21. These images are completely unfamiliar to the models and encompass various contexts, different lighting conditions, varying distances, etc.

FIGURE 21. Inference using the best global model of each architecture. The images were obtained from different internet sources.

In the first row, it can be observed that YOLOv6m and YOLO-NASm are the only models capable of detecting some relevant instances in the scene. However, they do not detect the same objects, and the confidence levels of the bounding boxes are not high. YOLOv6m and YOLOv8n only detect one instance each, demonstrating poor performance. Finally, YOLOv7x did not detect any instances, making it the worst-performing model for this scene.

The second row displays a distant column of smoke. Interestingly, none of the models were able to detect this instance, revealing their inability to detect objects in distant or panoramic scenes. Moving on to the third row, it depicts another scene that contains a column of smoke. Unlike the previous scene, this one has better lighting conditions and is evidently closer. In this scene, all models were able to detect the column of smoke, although they did not agree on the object's location. Furthermore, YOLOv6m placed three bounding boxes on the same object.

Moving to the fourth row, the scene showcases a distant column of smoke in a forest. Additionally, there is a helicopter with a bambi bucket. In this scene, YOLOv7x was unable to detect the column of smoke. The models YOLOv5s, YOLOv8n, and YOLO-NASm did detect the column of smoke but differed in the object's location. However, the confidence level of the bounding boxes was not high. The most notable result comes from the YOLOv6m model. It detected a column of smoke on the left side of the scene but also mistakenly identified the bambi bucket as fire. This highlights the incapability of the YOLOv6m model to detect objects in distant scenes, as it was the only model that made this error.

Finally, in the fifth row, there is a prominent column of smoke along with some small flames. The first evident observation is that the YOLOv5s model was unable to detect any instances. The YOLOv6m and YOLOv8n models detected the column of smoke, but with relatively low confidence levels. The YOLOv7x and YOLO-NASm models were able to detect both the column of smoke and the small flames; however, the confidence levels of the bounding boxes were low.

G. LIMITATIONS

Despite the valuable contributions of this study, it is important to acknowledge its limitations. Firstly, the evaluation was conducted using a single dataset, which may not fully represent the wide range of scenarios and variations encountered in real-world wildfire detection applications. Therefore, the generalizability of the findings to other datasets or environments should be approached with caution. Future studies should aim to incorporate diverse datasets to ensure a comprehensive understanding of the performance of these architectures across different contexts and conditions.

Secondly, it is worth noting that the evaluation was performed using pre-recorded images rather than real-time scenarios, such as those encountered in drone or live camera-based wildfire detection systems. The performance of the architectures in these dynamic and time-sensitive scenarios could potentially differ from the results obtained in this study. Therefore, additional research is needed to assess the real-time performance of the architectures and their ability to handle the challenges posed by rapidly changing scenes and limited processing resources.

Lastly, while this study provides valuable insights into the comparative performance of different architectures, it is essential to recognize that the field of deep learning is continuously evolving. Newer architectures and techniques may emerge that could outperform the models evaluated in this study. Therefore, it is crucial to stay updated with the latest advancements in the field and continue exploring novel approaches for improved wildfire detection systems.

IV. CONCLUSION AND FUTURE WORKS

In this paper, we conducted a comprehensive performance evaluation of various YOLO architectures, namely YOLOv5, YOLOv6, YOLOv7, YOLOv8, and YOLO-NAS, for smoke and wildfire detection. Our objective was to assess their effectiveness in addressing the challenges associated with early detection of wildfires. To achieve this, we utilized a well-curated dataset known as the Foggia dataset, which consists of 8,974 images specifically designed for smoke and wildfire detection.

The evaluation of the architectures was carried out using a set of metrics that included Recall, Precision, F1 score, and mean Average Precision (mAP). These metrics allowed us to perform a holistic evaluation of the models' performance in accurately detecting smoke and wildfire instances. By employing these metrics, we aimed to gain comprehensive insights into the unique features and limitations of each YOLO architecture in the context of smoke and wildfire detection.

To guarantee a comprehensive evaluation, we implemented a solid training and testing strategy. This encompassed a static count of 300 epochs, with an emphasis on enhancing recall for its application in detecting wildfires and smoke. Constant monitoring of model efficacy on validation sets was undertaken, culminating in an analysis on a bias-free test set, meticulously compiled from a variety of independent online sources. Such a methodology allowed us to establish a solid foundation for comparing the performance of different deep learning architectures in smoke and wildfire detection, facilitating informed decision-making and meaningful conclusions.

Throughout the experiments, the architectures that demonstrated a better balance across all metrics were YOLOv5, YOLOv7, and YOLOv8, both in validation and testing. YOLOv6 showed lower performance in terms of recall during validation but exhibited a good balance when evaluated on testing. However, for critical applications that require high recall to minimize false negatives, the YOLO-NAS variants stood out by achieving the highest recall scores among all models. This could be particularly beneficial in applications such as wildfire detection, where failing to identify a real

fire could have severe consequences. It is important to note, though, that the YOLO-NAS variants also exhibited lower precision performance in both validation and testing.

Regarding the visual results, it was observed that all top-performing models were able to identify the majority of relevant instances in the test set. However, when evaluated with new images obtained from the internet, they all displayed a combination of strengths and limitations.

Specifically, the models showed limited performance when the scenes were distant or had poor lighting conditions. Also, some models detected incorrect objects, mistaking them for instances of fire and smoke where they were not present. In scenes with favorable lighting conditions and closer proximity, the models exhibited good performance as they were able to correctly identify multiple important instances.

In summary, when evaluating different architectures and variants for wildfire and smoke detection, it becomes apparent that there is no ''best'' model that excels in all aspects. The choice of model will depend on the specific needs of the application, considering accuracy, recall, inference time, and the balance between these parameters.

For future work, it would be beneficial to extend this analysis to other datasets, further expanding our understanding of these models in different contexts and under diverse conditions. This would contribute to a broader knowledge base and enhance the generalizability of the findings. Additionally, given the dynamic nature of deep learning and the continuous advancements in the field, there is ample opportunity to build upon these findings and explore novel techniques to enhance the performance of the architectures. Future work can delve into fine-tuning the architectures, optimizing hyperparameters, or even investigating ensemble methods to further boost detection accuracy and robustness.

Overall, this study not only advances the field of computer vision in smoke and wildfire detection but also serves as a foundation for future research endeavors aimed at improving the effectiveness of detection systems and minimizing the devastating impact of wildfires. With these findings as a foundation, researchers can now delve deeper, propose innovative modifications, and push the boundaries of this field, ultimately leading to more effective and efficient wildfire detection systems.

REFERENCES

[1] P. Xofis, G. Tsiourlis, and P. Konstantinidis, ''A fire danger index for the early detection of areas vulnerable to wildfires in the eastern Mediterranean region,'' *Euro-Medit. J. Environ. Integr.*, vol. 5, no. 2, p. 32, Aug. 2020.

[2] F.-N. Robinne, D. W. Hallema, K. D. Bladon, and J. M. Buttle, ''Wildfire impacts on hydrologic ecosystem services in North American high-latitude forests: A scoping review,'' *J. Hydrol.*, vol. 581, Feb. 2020, Art. no. 124360.

[3] D. Wang, D. Guan, S. Zhu, M. M. Kinnon, G. Geng, Q. Zhang, H. Zheng, T. Lei, S. Shao, P. Gong, and S. J. Davis, ''Economic footprint of California wildfires in 2018,'' *Nature Sustainability*, vol. 4, no. 3, pp. 252–260, Dec. 2020.

[4] D. A. Jaffe, S. M. O'Neill, N. K. Larkin, A. L. Holder, D. L. Peterson, J. E. Halofsky, and A. G. Rappold, ''Wildfire and prescribed burning impacts on air quality in the United States,'' *J. Air Waste Manage. Assoc.*, vol. 70, no. 6, pp. 583–615, Jun. 2020.

[5] J. E. Halofsky, D. L. Peterson, and B. J. Harvey, ''Changing wildfire, changing forests: The effects of climate change on fire regimes and vegetation in the Pacific Northwest, USA,'' *Fire Ecol.*, vol. 16, no. 1, pp. 1–26, Dec. 2020.

[6] P. F. Hessburg, S. Charnley, A. N. Gray, T. A. Spies, D. W. Peterson, R. L. Flitcroft, K. L. Wendel, J. E. Halofsky, E. M. White, and J. Marshall, ''Climate and wildfire adaptation of inland northwest U.S. forests,'' *Frontiers Ecol. Environ.*, vol. 20, no. 1, pp. 40–48, Feb. 2022.

[7] E. D. Ponte, F. Alcasena, T. Bhagwat, Z. Hu, L. Eufemia, A. P. D. Turetta, M. Bonatti, S. Sieber, and P.-L. Barr, ''Assessing wildfire activity and forest loss in protected areas of the Amazon basin,'' *Appl. Geography*, vol. 157, Aug. 2023, Art. no. 102970.

[8] S. Mansoor, I. Farooq, M. M. Kachroo, A. E. D. Mahmoud, M. Fawzy, S. M. Popescu, M. N. Alyemeni, C. Sonne, J. Rinklebe, and P. Ahmad, ''Elevation in wildfire frequencies with respect to the climate change,'' *J. Environ. Manage.*, vol. 301, Jan. 2022, Art. no. 113769.

[9] D. Kinaneva, G. Hristov, G. Georgiev, P. Kyuchukov, and P. Zahariev, ''An artificial intelligence approach to real-time automatic smoke detection by unmanned aerial vehicles and forest observation systems,'' in *Proc. Int. Conf. Biomed. Innov. Appl. (BIA)*, Sep. 2020, pp. 133–138.

[10] S. Jazebi, F. de León, and A. Nelson, ''Review of wildfire management techniques—Part II: Urgent call for investment in research and development of preventative solutions,'' *IEEE Trans. Power Del.*, vol. 35, no. 1, pp. 440–450, Feb. 2020.

[11] D. Kinaneva, G. Hristov, J. Raychev, and P. Zahariev, ''Early forest fire detection using drones and artificial intelligence,'' in *Proc. 42nd Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2019, pp. 1060–1065.

[12] K. Bot and J. G. Borges, ''A systematic review of applications of machine learning techniques for wildfire management decision support,'' *Inventions*, vol. 7, no. 1, p. 15, Jan. 2022.

[13] P. Sharma, S. Gupta, S. Vyas, and M. Shabaz, ''Retracted: Object detection and recognition using deep learning-based techniques,'' *IET Commun.*, vol. 17, no. 13, pp. 1589–1599, Aug. 2023.

[14] G. Jocher, ''YOLOv5 by ultralytics,'' Ultralytics, Los Angeles, CA, USA, May 2020.

[15] L. Ting, Z. Baijun, Z. Yongsheng, and Y. Shun, ''Ship detection algorithm based on improved YOLO v5,'' in *Proc. 6th Int. Conf. Autom., Control Robot. Eng. (CACRE)*, Jul. 2021, pp. 483–487.

[16] J. Zhang, J. Zhang, K. Zhou, Y. Zhang, H. Chen, and X. Yan, ''An improved YOLOv5-based underwater object-detection framework,'' *Sensors*, vol. 23, no. 7, p. 3693, Apr. 2023.

[17] M. L. Mekhalfi, C. Nicolò, Y. Bazi, M. M. A. Rahhal, N. A. Alsharif, and E. A. Maghayreh, ''Contrasting YOLOv5, transformer, and EfficientDet detectors for crop circle detection in desert,'' *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[18] Z. Liu, X. Gao, Y. Wan, J. Wang, and H. Lyu, ''An improved YOLOv5 method for small object detection in UAV capture scenes,'' *IEEE Access*, vol. 11, pp. 14365–14374, 2023.

[19] W. Liu, K. Quijano, and M. M. Crawford, ''YOLOv5-tassel: Detecting tassels in RGB UAV imagery with improved YOLOv5 based on transfer learning,'' *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 8085–8094, 2022.

[20] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, and X. Wei, ''YOLOv6: A single-stage object detection framework for industrial applications,'' 2022, *arXiv:2209.02976*.

[21] C. Gupta, N. S. Gill, P. Gulia, and J. M. Chatterjee, ''A novel finetuned YOLOv6 transfer learning model for real-time object detection,'' *J. Real-Time Image Process.*, vol. 20, no. 3, p. 42, Jun. 2023.

[22] J. Barbosa, R. Graça, G. Santos, and M. J. M. Vasconcelos, ''Automatic analogue gauge reading using smartphones for industrial scenarios,'' in *Proc. 8th Int. Conf. Mach. Learn. Technol.*, Mar. 2023, pp. 277–284.

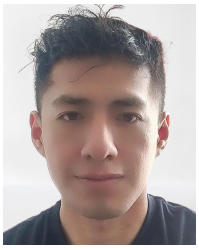
[23] B. Kang and C.-S. Jeong, ''ARTD-Net: Anchor-free based recyclable trash detection net using edgeless module,'' *Sensors*, vol. 23, no. 6, p. 2907, Mar. 2023.

[24] L. Huang, W. Huang, H. Gong, C. Yu, and Z. You, ''PEFNet: Position enhancement faster network for object detection in roadside perception system,'' *IEEE Access*, vol. 11, pp. 73007–73023, 2023.

[25] A. A. Adegun, J. V. F. Dombeu, S. Viriri, and J. Odindi, ''State-of-the-Art deep learning methods for objects detection in remote sensing satellite images,'' *Sensors*, vol. 23, no. 13, p. 5849, Jun. 2023.

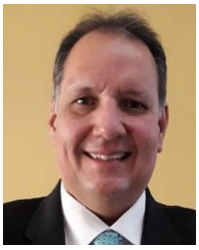
[26] M. Hussain, "YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, no. 7, p. 677, Jun. 2023.
[27] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2023, pp. 7464–7475.
[28] M. Yasir, L. Zhan, S. Liu, J. Wan, M. S. Hossain, A. T. I. Colak, M. Liu, Q. U. Islam, S. R. Mehdi, and Q. Yang, "Instance segmentation ship detection based on improved YOLOv7 using complex background SAR images," *Frontiers Mar. Sci.*, vol. 10, May 2023, Art. no. 1113669.
[29] K. Patel, C. Bhatt, and P. L. Mazzeo, "Improved ship detection algorithm from satellite images using YOLOv7 and graph neural network," *Algorithms*, vol. 15, no. 12, p. 473, Dec. 2022.
[30] M. Hussain, H. Al-Aqrabi, M. Munawar, R. Hill, and T. Alsboui, "Domain feature mapping with YOLOv7 for automated edge-based pallet racking inspections," *Sensors*, vol. 22, no. 18, p. 6927, Sep. 2022.
[31] I. Gallo, A. U. Rehman, R. H. Dehkordi, N. Landro, R. La Grassa, and M. Boschetti, "Deep object detection of crop weeds: Performance of YOLOv7 on a real case dataset from UAV images," *Remote Sens.*, vol. 15, no. 2, p. 539, Jan. 2023.
[32] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by ultralytics," Ultralytics, Los Angeles, CA, USA, Jan. 2023.
[33] J.-H. Kim, N. Kim, and C. S. Won, "High-speed drone detection based on YOLO-V8," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–2.
[34] A. Vats and D. C. Anastasiu, "Enhancing retail checkout through video inpainting, YOLOv8 detection, and DeepSort tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 5529–5536.
[35] S. Akhtar, M. Hanif, and H. Malih, "Automatic urine sediment detection and classification based on YOLOv8," in *Proc. Int. Conf. Comput. Sci. Appl.*, O. Gervasi, B. Murgante, A. M. A. C. Rocha, C. Garau, F. Scorza, Y. Karaca, and C. M. Torre, Eds. Cham, Switzerland: Springer, 2023, pp. 269–279.
[36] A. Aboah, B. Wang, U. Bagci, and Y. Adu-Gyamfi, "Real-time multi-class helmet violation detection using few-shot data sampling technique and YOLOv8," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 5349–5357.
[37] K. Xia, Z. Lv, C. Zhou, G. Gu, Z. Zhao, K. Liu, and Z. Li, "Mixed receptive fields augmented YOLO with multi-path spatial pyramid pooling for steel surface defect detection," *Sensors*, vol. 23, no. 11, p. 5114, May 2023.
[38] S. Tamang, B. Sen, A. Pradhan, K. Sharma, and V. K. Singh, "Enhancing COVID-19 safety: Exploring YOLOv8 object detection for accurate face mask classification," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, pp. 892–897, Feb. 2023.
[39] Y. Liu, Y. Sun, B. Xue, M. Zhang, G. G. Yen, and K. C. Tan, "A survey on evolutionary neural architecture search," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 2, pp. 550–570, Feb. 2023.
[40] J. Xu and Y. Lu, "OpenWeedGUI: An open-source graphical user interface for weed imaging and detection," *Proc. SPIE*, vol. 12539, Jun. 2023, Art. no. 1253909.
[41] X. Chu, L. Li, and B. Zhang, "Make RepVGG greater again: A quantization-aware approach," 2022, *arXiv:2212.01593*.
[42] B. Liberatori, C. A. Mami, G. Santacatterina, M. Zullich, and F. A. Pellegrino, "YOLO-based face mask detection on low-end devices using pruning and quantization," in *Proc. 45th Jubilee Int. Conv. Inf., Commun. Electron. Technol. (MIPRO)*, May 2022, pp. 900–905.
[43] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jul. 2020, pp. 237–242.
[44] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics*, vol. 10, no. 3, p. 279, Jan. 2021.
[45] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," *Electronics*, vol. 9, no. 3, p. 537, Mar. 2020.
[46] H. Zhu, H. Wei, B. Li, X. Yuan, and N. Kehtarnavaz, "A review of video object detection: Datasets, metrics and methods," *Appl. Sci.*, vol. 10, no. 21, p. 7834, Nov. 2020.
[47] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A comprehensive survey of loss functions in machine learning," *Ann. Data Sci.*, vol. 9, no. 2, pp. 187–212, Apr. 2022.
[48] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2888–2897.
[49] S. Wu, J. Yang, X. Wang, and X. Li, "IoU-balanced loss functions for single-stage object detection," *Pattern Recognit. Lett.*, vol. 156, pp. 96–103, Apr. 2022.
[50] S. Du, B. Zhang, and P. Zhang, "Scale-sensitive IOU loss: An improved regression loss function in remote sensing object detection," *IEEE Access*, vol. 9, pp. 141258–141272, 2021.
[51] X. Li, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Dec. 2020, pp. 21002–21012.

EDMUNDO CASAS (Member, IEEE) received the B.Sc. degree in electronic engineering, the B.Sc. degree in electronic civil engineering, and the master's degree in electronic civil engineering from Universidad Técnica Federico Santa María, Chile, in 2002, 2004, and 2007, respectively. He was a Professor and a Researcher with Chilean universities, such as Universidad Técnica Federico Santa María; Universidad de Chile; Pontificia Universidad Católica de Chile; Universidad Adolfo Ibáñez; Universidad Diego Portales; Tecnológico de Monterrey, Mexico; Universidad de EAFIT, Colombia; and the University of San Francisco, USA. He studied innovation and technology transfer with Stanford University, in 2013; entrepreneurship and investment with the University of California at San Diego, in 2014; and applied artificial intelligence with The University of Chicago, USA, in 2018. In 2020, he was with ESE, PADE: Senior Business Management Program, Universidad de Los Andes, Chile. In the workplace, he was with Endesa Chile (currently ENEL) in electricity generation, from 2002 to 2005, and then with SONY Electronics, from 2005 to 2007. He founded the company Kauel Inc., since 2007, and the company focused on artificial intelligence applied in computer vision for Energy Industry. He is currently a Board Member with Dual Vision Cognitive, an artificial intelligence company, and Kyon, a virtual reality company. He is a member of Endeavor as a Prominent Entrepreneur, a member of the Council with the Advanced Center for Electricity and Electronics, and an Advisor in digital transformation with Universidad Técnica Federico Santa María. He is a member of the Society of Petroleum Engineers. He is part of Fuel and Power Research Network and Texas Oil & Gas Association. He received the Prize as a Student for the Best Mathematician and Chess Player in Chile, in 1996, the Prize for the Best Invention of the Presidency of the Republic of Chile, in 2010, and the Prize of the 100 Young Leaders of Chile by Adolfo Ibáñez University and El Mercurio, in 2010, and he was selected and awarded as the CEO together with Kauel Inc., as one of the leading growth companies by FORBES, in 2022.



LEO RAMOS (Student Member, IEEE) received the degree (magna cum laude) in information technology engineering with a focus on artificial intelligence from Yachay Tech University, Ecuador, in 2023. He is currently pursuing the Ph.D. degree in computer science with the Universitat Autònoma de Barcelona, Spain. He was a Research Assistant under the supervision of distinguished researchers on multiple occasions. He is also working as a Senior Research Engineer at Zeus Intelligent Solutions, USA, where he is engaged in the research and development of machine learning and deep learning models for various applications, with a special focus on computer vision. He is currently a Researcher with Sleep Care Clinics, U.K., conducting research in artificial intelligence for medical data analysis. In addition, he is a Trainee Engineer with Kauel Inc., USA, contributing to the development of cutting-edge intelligent systems for the energy sector. He collaborates with renowned researchers from Venezuela, Spain, France, U.K., and USA, including ongoing collaborations with NASA scientists. He has broadened his skill set and network through internships in both industry and academia, nationally and internationally. His research interests include deep learning, computer vision, natural language processing, and data science.

EDUARDO BENDEK received the master's degree in mechanical engineering and minor in astrophysics and the Ph.D. degree in optical sciences. He is currently a Chilean Scientist and an Engineer with the Jet Propulsion Laboratory, NASA, where he designs instruments for space telescopes. In his laboratory, he develops technologies for planet detection. In addition, he is an Advisor with the NASA's Planetary Exploration Program. Beyond his academic achievements, he has entrepreneurial experience as the Co-Founder of YxWireless and serves on the board of several companies. He is the author of more than 100 scientific publications and holds multiple patents. His advancements led him to be honored with the NASA's Exceptional Technology Achievement Medal, in 2015.



FRANCKLIN RIVAS-ECHEVERRÍA (Senior Member, IEEE) received the B.S. degree in systems engineering, the M.S. degree in control engineering, the Ph.D. degree in applied science, and the B.S. degree in law from Universidad de Los Andes, Venezuela, in 1993, 1996, 2000, and 2017, respectively, and the master's degree in artificial intelligence and machine learning from Purdue University, USA, in 2023. He was a full-time Professor and a Researcher with Universidad de Los Andes, from 1994 to 2018; Universidad Técnica Federico Santa María, Chile, from 2018 to 2022; and Yachay Tech University, Ecuador, from 2022 to 2023. He is currently a Professor and a Researcher with Pontificia Universidad Católica del Ecuador Sede Ibarra, Ecuador, and MIU City University Miami, USA. He is also the Chief Research Officer with Kauel Inc., USA. His research interests include artificial intelligence and data science applications. He received several international and national awards, including recognition awarded by ''Revista Gerente'' as one of the 100 most successful managers in Venezuela, Halliburton awarded him a recognition for contributions and dedication to the development of petroleum technology, the Outstanding Professor Award by Universidad Técnica Federico Santa María, in 2021, the Outstanding Leadership Award from the Internet 2.0 Conference, in 2023, and the Bicentennial Distinction by Universidad de Los Andes, in 2023.