IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 February 2024, accepted 13 March 2024, date of publication 19 March 2024, date of current version 26 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3378568

SURVEY

YOLOv1 to v8: Unveiling Each Variant–A Comprehensive Review of YOLO

MUHAMMAD HUSSAIN
Department of Computer Science, University of Huddersfield, HD1 3DH Huddersfield, U.K.
e-mail: m.hussain@hud.ac.uk

ABSTRACT This paper implements a systematic methodological approach to review the evolution of YOLO variants. Each variant is dissected by examining its internal architectural composition, providing a thorough understanding of its structural components. Subsequently, the review highlights key architectural innovations introduced in each variant, shedding light on the incremental refinements. The review includes benchmarked performance metrics, offering a quantitative measure of each variant's capabilities. The paper further presents the performance of YOLO variants across a diverse range of domains, manifesting their real-world impact. This structured approach ensures a comprehensive examination of YOLOs journey, methodically communicating its internal advancements and benchmarked performance before delving into domain applications. It is envisioned, the incorporation of concepts such as federated learning can introduce a collaborative training paradigm, where YOLO models benefit from training across multiple edge devices, enhancing privacy, adaptability, and generalisation.

INDEX TERMS Computer vision, YOLO, edge-computing, manufacturing, object detection, realtime.

I. INTRODUCTION

In the realm of computer vision [1], [2], [3], the imperative task of object detection has undergone a paradigmatic evolution [4], [5], [6], catalysed by the revolutionary advent of the You Only Look Once (YOLO) architecture in 2016 [7]. YOLOs innovative approach diverged from the traditional two-stage object detection architectures, proposing a unified architecture with the ability to predict bounding boxes and class probabilities concurrently, all within the exigencies of real-time processing [8]. The prominence of YOLO rests not merely in its single-stage approach but in its iterative progression, evolving from its nascent version, YOLOv1, to the present variant, YOLOv8. Each iteration manifested a confluence of innovative architectural advancements, affording YOLO an unprecedented adeptness and versatility across a plethora of industrial applications. YOLOs ethos is underscored by an intrinsic equilibrium between accuracy and speed, making it a feasible proposition to the research community and industry practitioners alike.

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

This article embarks aims to untangle the intricacies underpinning YOLOs variants, commencing with a meticulous analysis of the architectural advancements from YOLOv1 to YOLOv8. Exploring the rationale behind these structural adaptations, their mutual relationships, and their consequential impact on the efficacy of object detection forms the bedrock of this paper. Transcending architectural intricacies, the paper examines the core of YOLOs effectiveness, which lies in its training methodologies. Through the amalgamation of data augmentation, transfer learning principles, and strategic implementation of internal architectural enhancements, YOLO has fostered a robustness that extends beyond the limitations of domain-specific peculiarities.

Diving deeper into the deployment realm, the paper examines the real-world impact of YOLO. Through specific examples where YOLO has showcased its efficacy, we present insights into YOLOs evolution from a baseline architecture to a transformative asset in real-world applications. However, as YOLO ascends penetrates into new domains, it encounters its own set of challenges. From navigating through occlusions to efficiently managing scale variations and fine-grained object detection scenarios [9], YOLO faces a myriad of obstacles. By acknowledging these challenges, we initiate

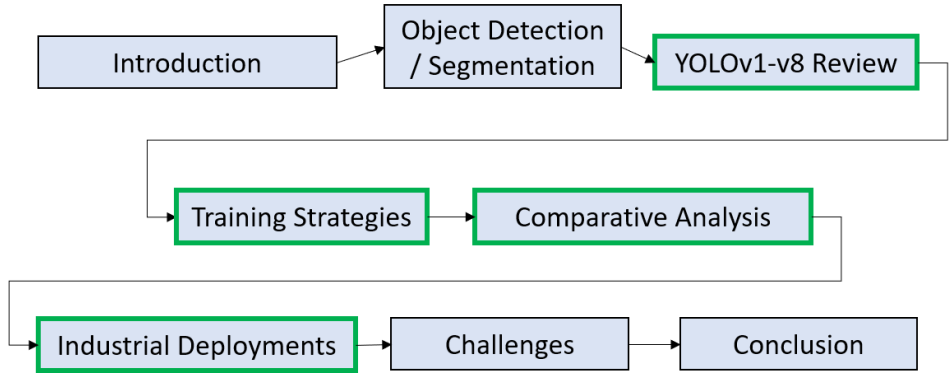

FIGURE 1. Visual structure of this review.

a dialogue on the future trajectory of research, outlining potential pathways to strengthen YOLOs robustness in the realm of object detection.

In summary, this article dives into the evolving YOLO architectures, comprehensively evaluating their effectiveness, and pondering over future prospects. The article unravels how YOLO has transformed over the epochs of time, how effective it is now, and what the future of YOLO variants may be in the domain of computer vision.

A. SURVEY OBJECTIVE

This article seeks to examine the factors fuelling the profound adoption of the YOLO variants, with a focus on its evolution from YOLOv1 to YOLOv8. Figure 1 presents the article structure with the key components of the article highlighted in green. These components form the key objectives of this paper:

1) **Architectural Evolution Analysis:** Examine the architectural innovations across YOLO variants, elucidating motivations and impact on real-time industrial applications.
2) **Training Strategies Scrutiny:** Analyse YOLO's training methodologies, including data augmentation and transfer learning, to understand its adaptability across diverse domains.
3) **Real-world Impact Assessment:** Explore specific domains where YOLO has manifested impressive efficacy, showcasing its practical versatility.
4) **Challenges and Future Directions Exploration:** Identify realtime challenges, such as occlusions and scale variations, and propose future research directions to fortify YOLO's standing in object detection.

B. IMPORTANCE OF SURVEY

Although several papers have reviewed YOLO architectures, they often exhibit limitations such as focusing on specific YOLO variants [10], [11], or concentrating on particular application domains [12]. However, this review distinguishes itself as the first to provide an in-depth analysis of mainstream YOLO variants from YOLOv1 to YOLOv8. The analysis delves into the innovations fueling the performance of each variant, offering a comparative study that spans more than 20 domains.

Furthermore, beyond examining the strengths of YOLO architectures, this comprehensive review sheds light on the persistent challenges faced by the YOLO series. By outlining current limitations and areas for improvement, the review aims to present a nuanced understanding of the ongoing hurdles.

Additionally, it anticipates future developments and enhancements, providing insights into potential directions for overcoming existing challenges. This forward-looking approach positions the review as a valuable resource not only for understanding the historical evolution of YOLO but also for anticipating its trajectory in addressing emerging issues and meeting the demands of diverse domains.

C. ORGANIZATION OF PAPER

This article is structured to succinctly examine the evolution and inspiration fuelling the popularity of YOLO variants in industrial applications. Beginning with an introduction that lays the foundations, subsequent sections are intricately structured. Section II presents an overview of object detection. Section III, delves into the motivations and implications of architectural reforms across the variants, YOLOv1 to YOLOv8.

Section IV, scrutinizes the versatility of YOLO variants through an examination of training methodologies, including data augmentation, transfer learning, and training datasets. In Section V, a rigorous empirical assessment of YOLOv1-v8 is conducted, benchmarking against contemporaneous models to quantify performance with respect to Mean-Average Precision (MAP), Frames Per Second (FPS) and internal intricacies such as nature of loss functions deployed.

Section VI, explores wide-ranging industrial applications where YOLO has demonstrated efficacy, showcasing its practical versatility. Section VII, identifies barriers like handling occlusions, addressing biases, real-time edge deployment and proposes future research directions.

Finally, Section VIII, summarises key findings, highlighting factors contributing towards YOLO's popularity and its significant implications for the field of object detection. This organized structure ensures a coherent and insightful journey through the multifaceted analysis of YOLO's evolution and impact in object detection and the wider field of computer vision.

II. OBJECT DETECTION

Addressing the intricacies of object detection presents numerous challenges. A key issue involves effectively managing fluctuations in image resolutions and aspect ratios [13], a task aggravated when the target objects manifest substantial differences in spatial dimensions [14]. The presence of class imbalance, particularly in scenarios where ascertain a sufficient number of images for specific classes is challenging [15], can detrimentally impact architectural performance, leading to biased predictions [16].

Furthermore, a noteworthy hurdle is the computational complexity associated with object detection architectures, demanding considerable computational resources in terms of power, memory, and time [17], [18]. Figure 2, illustrates object detection for both single and multiple objects in an image, detectors with deep internal networks require significant computational capabilities to process intricate datasets and extract essential features.

Object detection can be bifurcated into two categories: single- and two-stage detectors. The latter contains proposing candidate regions within an image, followed by classification and localization of proposed regions. Examples of two-stage detectors include RCNN (Region-based Convolutional Neural Network) [19], Fast R-CNN [20], Faster R-CNN [21], and FPN (Feature Pyramid Network) [22].

RCNN [19], proposed in 2014, deployed selective search for candidate region proposals, utilising a convolutional network for feature extraction. Fast R-CNN [20] facilitates these concerns by proposing ROI pooling, which significantly reduces computations by extracting fixed-size feature maps for each region from the original feature maps.

Faster R-CNN [21] enriched upon Fast R-CNN by implementing the Region Proposal Network (RPN). This innovation eradicated the need for a separate proposal stage by directly generating region proposals from feature maps, optimising both speed and accuracy.

FPN (Feature Pyramid Network) [22] tackled the challenge of detecting targets at multiple scales by generating a feature pyramid. This pyramid fused feature maps of varying resolutions from different network stages, empowering effective detection of targets across different scales. Notwithstanding their impressive accuracy, two-stage detectors, are limited by their high computational demands.

In contrast, single-stage detectors aim to detect objects in a single pass, side-stepping the need for a separate region proposal step. Notable single-stage detectors include SSD (Single Shot Multibox Detector), YOLO variants (You Only Look Once), RefineDet++, DSSD (Deconvolution Single Shot Detector), and RetinaNet.

SSD [23] deploys manifold convolutional feature maps at various scales to predict bounding boxes and class probability scores, effectively detecting objects of various sizes and shapes in a single forward pass.

RefineDet++ [24] optimises the original RefineDet architecture through iterative refinement of target proposals across multiple stages, improving accuracy via enhanced feature fusion mechanisms and refined target boundaries.

DSSD (Deconvolution Single Shot Detector) integrates deconvolution layers to preserve spatial information lost during feature pooling, enabling the model to capture fine-grained details by maintaining spatial resolution.

RetinaNet [25] addresses class imbalance via Focal Loss, attributing higher weights to misclassified samples, enhancing the architecture's ability to handle class imbalance and improve detection performance.

Vision Transformer (ViT) was introduced in 2020 [26]. Based the encoder and decoder mechanism, ViT continues the concept of tokens to visual data streams. As an alternate to CNNs, ViT can be utilized for the backbone feature extraction work. Selecting ResNet as a baseline, Wu et al. [27], integrated ViTs by replacing the ultimate convolutional layer. This enabled the preceding convolutional layers to extract low-level features, which then segued into the ViT, demonstrating the adaptability of the transformer architecture in the territory of computer vision.

III. EVOLUTION OF YOLO ARCHITECTURE:

A. YOLOv1

Announced in 2016, YOLOv1 marked a profound leap in single-shot object detection. Enthused by the GoogLeNet architecture [28], YOLOv1 deployed a unique approach by substituting GoogLeNet's inception modules with (1×1) convolution followed by (3×3) convolutional filters.

The architecture, benchmarked on the VOC Pascal Dataset 2007 and 2012 [29], exploited the Darknet framework for training. Featuring 24 convolution layers, with only four of which were followed by max-pooling layers, YOLOv1 embraced (1×1) convolutions and global average pooling as standout features.

Initially trained on the ImageNet dataset [30], the model was exposed to fine-tuning by adding four additional convolutional layers and two fully connected layers with randomly initialized weights. The activation function employed Leaky Rectified Linear Unit (LReLU), except for the last layer with a linear activation function. Despite its pioneering status, YOLOv1 exhibited drawbacks, including large localization errors and lower recall compared to two-stage object detectors.

B. YOLOv2

YOLOv2 [31] was inspired by the once popular VGG architecture, featuring the darknet-19 framework with 19 con-

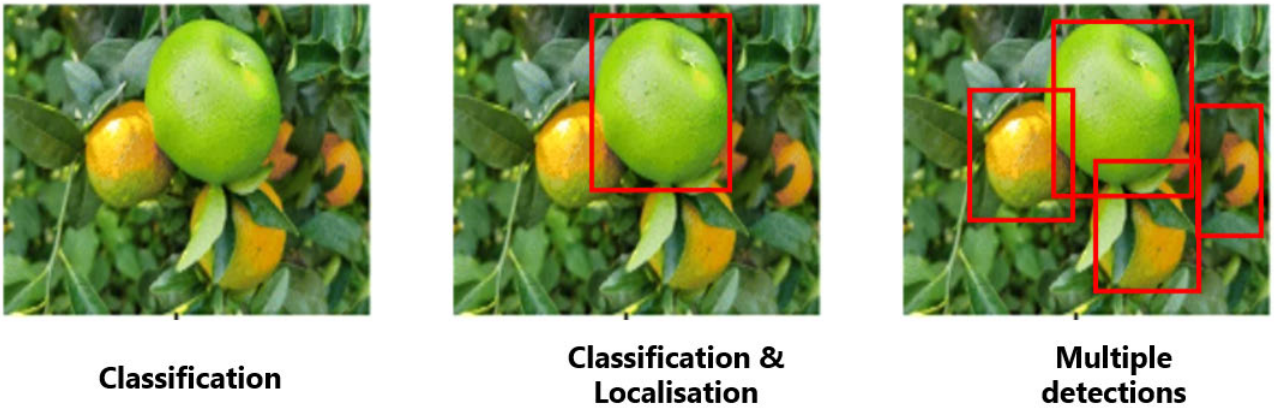

FIGURE 2. Single and multiple objects in an image: Classification, Localization, Segmentation.

TABLE 1. YOLOv2 darknet-19 framework.

Type	Filters	Size/Stride	Output
Convolutional	32	3 x 3	224 x 224
Maxpool	2 x 2/2	112 x 112	
Convolutional	64	3 x 3	112 x 112
Maxpool	2 x 2/2	56 x 56	
Convolutional	128	3 x 3	56 x 56
Convolutional	64	1 x 1	56 x 56
Convolutional	128	3 x 3	56 x 56
Maxpool	2 x 2/2	28 x 28	
Convolutional	256	3 x 3	28 x 28
Convolutional	128	1 x 1	28 x 28
Convolutional	256	3 x 3	28 x 28
Maxpool	2 x 2/2	14 x 14	
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Maxpool	2 x 2/2	7 x 7	
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	1000	1 x 1	7 x 7
Avgpool	Global	1000	
Softmax			

volutional layers and 5 max pooling layers as presented in Table 1. Prominent features included the integration of 1 × 1 convolutions for down sampling across the depth of the input volume. YOLOv2 incorporated several data augmentation techniques, such as random crops and rotations, for enhanced training. However, YOLOv2 faced challenges in detecting smaller-sized objects. The architecture introduced new optimisation procedures, including Batch Normalization, High Resolution Classifier, Convolution with Anchor Boxes, Dimension Clusters, Direct Location Prediction, Fine-Grained Features, and Multi-Scale Training. Key Innovations in YOLOv2 included:

a: BATCH NORMALIZATION

Addressing the matter of Internal Covariate Shift during the training of neural networks, YOLOv2 employed Batch Normalization to normalize the outputs of all hidden layers, guaranteeing consistent distribution of weight matrices across different layers. This not only reduced superfluous shifts in deeper hidden layers but also acted as a regularisation technique. The use of Batch Normalization contributed to an approximately 2% increase in mean Average Precision (mAP).

b: HIGH RESOLUTION CLASSIFIER

In YOLOv2, the High Resolution Classifier was introduced to train the model on 448 × 448 sized images for classification before fine-tuning for object detection. This framework improved the model's ability to learn classification and adapt to high-resolution inputs, resulting in an approximately 4% increase in mAP.

c: CONVOLUTION WITH ANCHOR BOXES

Replacing the fully connected layer from the base variant, YOLOv2 employed convolution with anchor boxes leading to an approximately 7% increase in recall, albeit with a 0.3% reduction in mAP.

d: DIMENSION CLUSTERS

Moving away from pre-defined anchor boxes, YOLOv2 extracted anchor boxes using KMEANS [32] clustering, providing enhanced anchor boxes for optimised model training and performance.

e: DIRECT LOCATION PREDICTION

To address issues with pre-defined priors and model instability during bounding box prediction, YOLOv2 predicted location coordinates relative to grid-cell locations. This feature contributed to a 5% increase in mAP.

f: FINE-GRAINED FEATURES

Recognizing the challenge of effectively detecting smaller objects, YOLOv2 introduced fine-grained features by concatenating higher resolution features with lower resolution features via skip connections, resulting in a 1% increase in mAP.

g: MULTI-SCALE TRAINING

With the omission of fully connected layers, YOLOv2 operated on dimensions ranging from 320 × 320 to 608 × 608. The architecture randomly selected new dimensions, multiples of 32, providing flexibility and contributing to the model's ability to predict on various dimensions.

C. YOLOv3

YOLOv3 [33], addressed the shortcomings observed in its predecessors by concentrating on rectifying localisation errors and optimising detection efficiency, particularly for smaller objects. Benchmarked on the COCO dataset [34], YOLOv3 presented improved performance in detecting smaller objects, while encountering difficulties in achieving precise results for medium and large-sized objects.

Constructed on the Darknet-53 framework, YOLOv3 employs a robust network comprising of 53 convolutional layers, incorporating 3 × 3 and 1 × 1 convolutional filters along with skip connections, as presented in Table 2. Conspicuously, the Darknet-53 framework, with its 53 convolutional layers, achieved double the speed of ResNet-152 [35].

TABLE 2. YOLOv3 internal architecture.

Layer	Filters	Size	Repeat	Output Size
Image	—	—	—	416 × 416
Conv	32	3 × 3 / 1	1	416 × 416
Conv	64	3 × 3 / 2	1	208 × 208
Conv	32	1 × 1 / 1	Conv × 1	208 × 208
Conv	64	3 × 3 / 1	Conv × 1	208 × 208
Residual	—	—	Residual × 1	208 × 208
Conv	128	3 × 3 / 2	1	104 × 104
Conv	64	1 × 1 / 1	Conv × 2	104 × 104
Conv	128	3 × 3 / 1	Conv × 2	104 × 104
Residual	—	—	Residual × 2	104 × 104
Conv	256	3 × 3 / 2	1	52 × 52
Conv	128	1 × 1 / 1	Conv × 8	52 × 52
Conv	256	3 × 3 / 1	Conv × 8	52 × 52
Residual	—	—	Residual × 8	52 × 52
Conv	512	3 × 3 / 2	1	26 × 26
Conv	256	1 × 1 / 1	Conv × 8	26 × 26
Conv	512	3 × 3 / 1	Conv × 8	26 × 26
Residual	—	—	Residual × 8	26 × 26
Conv	1024	3 × 3 / 2	1	13 × 13
Conv	512	1 × 1 / 1	Conv × 4	13 × 13
Conv	1024	3 × 3 / 1	Conv × 4	13 × 13
Residual	—	—	Residual × 4	13 × 13

YOLOv3 introduced a foundational generic architecture inspired by the Feature Pyramid Network (FPN) [22], integrating elements such as residual blocks, skip connections, and up-sampling. Key attributes of YOLOv3 included:

a: DARKNET-53 BACKBONE

The architecture of YOLOv3 was based on the Darknet-53 framework, employing 3 × 3 and 1 × 1 convolutional filters alongside shortcut connections. This architecture, consisted of 53 convolutional layers, serving as a robust base for efficient object detection.

b: FEATURE PYRAMID NETWORK (FPN) INSPIRED DESIGN

Drawing inspiration from FPN, YOLOv3 incorporated heuristics such as residual blocks, skip connections, and up-sampling into its internal architectural footprint. This approach enhanced the network's capacity to detect objects efficiently across varying scales.

c: THREE-SCALE DETECTION MECHANISM

YOLOv3 generated feature maps at three distinct scales, down-sampling the input at factors of 32, 16, and 8. Detection was carried out on a 13 × 13 feature map after a series of convolutions, followed by a 26 × 26 feature map obtained via up-sampling and concatenation. Additionally, a 52 × 52 feature map was involved in the detection process. This three-scale mechanism enabled YOLOv3 to detect large, medium, and small-sized objects using distinct feature maps.

D. YOLOv4

Authors of YOLOv4 [36], introduced a plethora of advanced techniques and sophisticated methodologies, distinguishing YOLOv4 as a faster and more accurate object detector tailored for production systems compared to its predecessors.

YOLOv4 architecture was defined through a sequence of pivotal components: initial image processing, feature extraction utilising potent networks like VGG16 [37], Darknet53, and ResNet50, feature scaling with neck structures like Feature Pyramid Network (FPN) and Path Aggregation Network (PAN) [38], and the integration of single-stage and two-stage detectors for prediction.

In their experimentation with architectures, authors compared CSPResNeXt50, CSPDarknet53, and EfficientNetB3, ultimately selecting CSPDarknet53 as the backbone. CSPDarknet53, featured 29 convolution layers with 3 × 3 filters and around 27.6 million parameters, incorporating Cross-stage partial connections (CSP) to enhance gradient combination efficiency with minimal computational cost. Key architectural components included:

a: SPATIAL PYRAMID POOLING (SPP)

To accommodate various input dimensions without resizing or reshaping, YOLOv4 integrated Spatial Pyramid Pooling (SPP) [35]. Sandwiched between a convolutional blocks and fully connected layers, SPP mapped any input size to a fixed-size output, facilitating object detection for images of varying sizes.

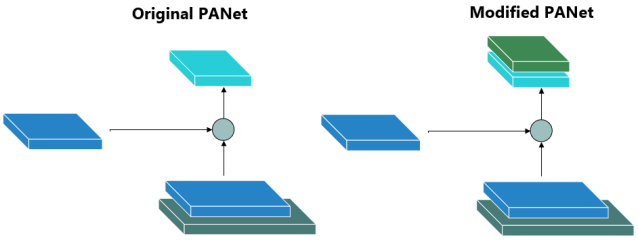

FIGURE 3. YOLOv4 path aggregation (a) Addition (b) Concatenation.

b: PATH AGGREGATION NETWORK (PAN)

In the traditional PANet, neighbouring layers were usually added together for mask predictions, employing adaptive feature pooling, as shown in Figure 3 (a). However, in YOLOv4, PANet directs a concatenation operation, combining the feature maps from different layers along a specific axis, creating a new feature map that contains information from both layers, as shown in Figure 3 (b). This modification aims to capture intricate details and context across multiple scales, enhancing the accuracy of predictions.

E. YOLOv5

YOLOv5 [39] epitomizes a paradigm shift as it transitions from the Darknet framework to PyTorch. The authors retained many improvements introduced in YOLOv4, in addition to notable changes. The architecture initiated a strided convolution layer with a large window size, aimed at reducing memory and computational costs. Subsequent convolutional layers extract pertinent features from the input image. The SPPF (spatial pyramid pooling fast) layer and subsequent convolution layers processed features at various scales, while up sample layers enhanced the resolution of feature maps. The SPPF layer accelerated computation by pooling features of different scales into a fixed-size feature map. Each convolutional layer was paired with batch normalization (BN) [40] and SiLU activation.

a: NECK AND HEAD – SPPF AND MODIFIED CSP-PAN

The neck of YOLOv5 implemented SPPF and a modified CSP-PAN, while the head structure resembled that of YOLOv3.

b: AUGMENTATIONS AND IMPROVED GRID SENSITIVITY

YOLOv5 introduced several augmentations, including Mosaic, copy-paste, random affine, MixUp, HSV augmentation, and random horizontal flip, along with augmentations from the albumentations package. The model also enhanced grid sensitivity for stability against runaway gradients.

c: MODEL VARIANTS

YOLOv5 is manifested in five variants to accommodate for various applications and hardware requirements: YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large). Each variant varies in the width and depth of the convolution modules. For instance,

TABLE 3. YOLOv5 internal variant comparison.

Model	Average Precision (@50)	Parameters	FLOPs
YOLO-v5s	55.8%	7.5M	13.2B
YOLO-v5m	62.4%	21.8M	39.4B
YOLO-v5l	65.4%	47.8M	88.1B
YOLO-v5x	66.9%	86.7M	205.7B

YOLOv5n and YOLOv5s are lightweight models suitable for low-resource devices, while YOLOv5x is optimized for high performance, albeit at the expense of speed.

d: PERFORMANCE AND VERSIONS

Benchmarked on the MS COCO dataset test-dev 2017, YOLOv5x achieved an AP of 66.9% compromising of 86.7 Million parameters. Whilst, YOLOv5s, at the other end of the spectrum, achieved an AP of 55.8% compromising of 7.5 Million parameters, as presented in Table 3.

e: OPEN SOURCE AND ACCESSIBILITY

YOLOv5 is open source and actively maintained by Ultralytics, boasting over 250 contributors and frequent updates. The model is known for its ease of use, training, and deployment. Ultralytics provides a mobile version for iOS and Android, along with various integrations for labelling, training, and deployment.

F. YOLOv6

Meituan Vision AI Department [41] introduced YOLOv6 in September 2022 [42], boasting several innovative features claiming to enhance both efficiency and accuracy. The architectural footprint, incorporated an efficient backbone featuring RepVGG/CSPStackRep blocks, a PAN (Path Aggregation Network) topology neck, and an efficient decoupled head with a hybrid-channel strategy [43].

Conspicuously, the paper introduced sophisticated quantisation strategies, implementing post-training quantisation and channel-wise distillation, resulting in detectors that are not only faster but also more accurate. YOLOv6 builds upon the successes of its predecessors, notably YOLOv5, surpassing previous state-of-the-art models in terms of accuracy and speed. Key Features include:

a: EFFICIENT BACKBONE – EfficientRep AND ENHANCED NECK

YOLOv6 introduced a new backbone called EfficientRep, based on RepVGG, which leveraged higher parallelism compared to previous YOLO backbones. The neck of the network utilised PAN enhanced with RepBlocks or CSPStackRep Blocks for larger models. The redesigned backbone and neck contributed to improved efficiency and adaptability.

b: TASK ALIGNMENT LEARNING APPROACH FOR LABEL ASSIGNMENT

The architecture adopts the Task Alignment Learning approach, enhancing label assignment. Additionally,

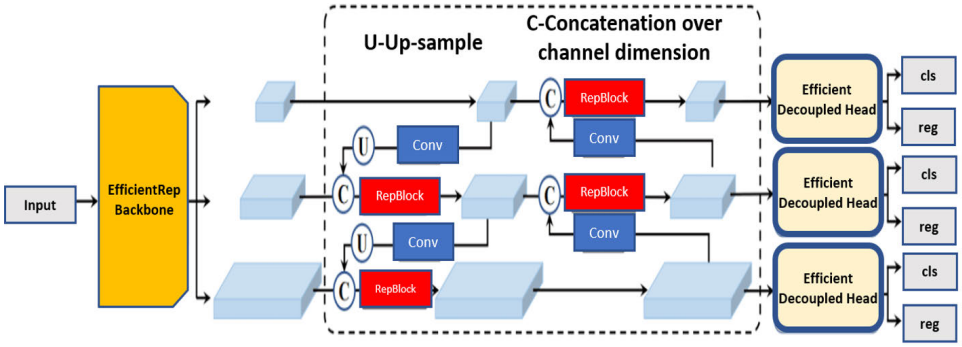

FIGURE 4. PANet configuration [2].

YOLOv6 incorporated new classification and regression losses, employing a classification VariFocal loss and an SIoU/GIoU regression loss.

c: SELF-DISTILLATION STRATEGY

YOLOv6 implemented a self-distillation strategy for both regression and classification tasks. This strategy assisted the model distill knowledge from its own predictions, contributing to improved performance and generalisation.

d: QUANTIZATION SCHEME WITH RepOptimiser AND CHANNEL-WISE DISTILLATION

The authors introduced a quantisation scheme for detection using RepOptimiser and channel-wise distillation. This scheme not only assisted in achieving a faster detector but also ensured that quantisation did not compromise accuracy.

e: BIDIRECTIONAL CONCATENATION (BiC) MODULE

YOLOv6 introduced a BiC module in the neck of the detector, enhancing localisation signals and delivering performance gains with negligible speed degradation.

f: ANCHOR-AIDED TRAINING (AAT) STRATEGY

AAT caters for both anchor-based and anchor-free paradigms without compromising inference efficiency.

g: ENHANCED BACKBONE AND NECK DESIGN

By deepening YOLOv6 to include another stage in the backbone and neck, the architecture achieved state-of-the-art performance on the COCO dataset at high-resolution input.

h: SELF-DISTILLATION STRATEGY

A new self-distillation strategy is implemented to boost the performance of smaller models of YOLOv6, enhancing the auxiliary regression branch during training and removing it at inference to avoid a marked speed decline.

i: MODEL VARIANTS AND PERFORMANCE

The authors provide eight scaled variants, ranging from YOLOv6-N to YOLOv6-L6, catering to different application requirements. Benchmarked on the MS COCO dataset test-dev 2017, the largest variant achieved an impressive AP of 57.2% while maintaining a speed of around 29 FPS on an NVIDIA Tesla T4.

G. YOLOv7

YOLOv7, released in 2022, represents an innovative advancement in the realm of object detection [43]. At the time of its release, it outperformed many present object detectors, ranging from 5 FPS to an impressive 160 FPS.

Notably, YOLOv7 was trained on the MS COCO dataset without leveraging pre-trained backbones, showcasing its ability to achieve remarkable results through its unique training approach. Architectural advents include:

a: EXTENDED EFFICIENT LAYER AGGREGATION NETWORK (E-ELAN)

YOLOv7 proposed an extended version of the efficient layer aggregation network (ELAN) [44], termed E-ELAN. ELAN is a strategic mechanism facilitating efficient learning and convergence in deep models by controlling the shortest longest gradient path. E-ELAN optimises this concept for models with unlimited stacked computational blocks. It achieves this by shuffling and merging cardinality features, thus augmenting the network's learning capabilities without compromising the original gradient path.

b: MODEL SCALING FOR CONCATENATION-BASED MODELS

YOLOv7 adopted a concatenation-based architecture, and to generate models of varying sizes, it introduced a novel mechanism for model scaling. Unlike standard scaling techniques, such as depth scaling, YOLOv7 ensured the depth and width of the block are scaled proportionally. This maintained the optimal structure of the model, preventing unwanted distortions in the hardware usage of the model.

c: PLANNED RE-PARAMETERIZED CONVOLUTION (RepConvN)

Inspired by re-parameterized convolutions (RepConv) from YOLOv6, YOLOv7 introduced RepConvN. In contrast to RepConv, RepConvN eradicates the identity connection,

TABLE 4. Key features and architectural evolution.

Variant	Year	Architecture
YOLOv1	2016	GoogLeNet-inspired
YOLOv2	2017	Darknet-19, VGG
YOLOv3	2018	Darknet-53
YOLOv4	2020	CSPDarknet53
YOLOv5	2020	CSPDarknet53
YOLOv6	2021	EfficientRep, CSPDarknet53
YOLOv7	2022	ELAN, CSPDarknet53
YOLOv8	2023	CSPDarknet53

addressing issues related to residual destruction in ResNet and concatenation in DenseNet.

d: COARSE AND FINE LABEL ASSIGNMENT

YOLOv7 employed a strategy of coarse label assignment for the auxiliary head and fine label assignment for the lead head. While the lead head is responsible for the final output, the auxiliary head assists with training, as presented in Figure 4.

e: BATCH NORMALIZATION IN CONV-BN-ACTIVATION

The integration of batch normalisation into conv-bn-activation involved incorporating the mean and variance of batch normalisation into the bias and weight of the convolutional layer during the inference stage.

H. YOLOv8

Ultralytics introduced YOLOv8 in January 2023 [45], signifying a significant evolution in the YOLO series by providing users with a comprehensive range of enhancements and versatile capabilities.

YOLOv8 introduces five scaled versions, catering to different application needs: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (extra large). Key Features include:

a: BACKBONE SIMILARITY TO YOLOv5 WITH C2F MODULE

YOLOv8 preserves a backbone architecture similar to YOLOv5, with profound adjustments in the CSPLayer, now referred to as the C2f module. This module, standing for ''cross-stage partial bottleneck with two convolutions,'' effectively combines high-level features with contextual information, contributing to enhanced detection accuracy.

b: SEMANTIC SEGMENTATION MODEL – YOLOv8-SEG

YOLOv8 extends its capabilities with a semantic segmentation model known as YOLOv8-Seg. It compromises of a CSPDarknet53 feature extractor followed by a C2F module, diverging from the conventional YOLO neck architecture. YOLOv8-Seg includes two segmentation heads responsible for predicting semantic segmentation masks for input images. YOLOv8 achieves state-of-the-art results in object detection and semantic segmentation benchmarks while maintaining efficiency.

TABLE 5. Training and optimization.

Variant	Framework	Dataset	Mechanisms
YOLOv1	Darknet	VOC Pascal Dataset	Grid-based approach
YOLOv2	Darknet	VOC Pascal, COCO Dataset	Hierarchical classification
YOLOv3	Darknet	COCO Dataset	Feature Pyramid Network
YOLOv4	Darknet	COCO Dataset	Enhanced quantization, PAN, RepVGG
YOLOv5	PyTorch	COCO Dataset	AutoAnchor, Mosaic, MixUp, EfficientNet
YOLOv6	PyTorch	COCO	RepVGG, PAN, EfficientRep
YOLOv7	Darknet	COCO Dataset	ELAN, Model scaling
YOLOv8	PyTorch	COCO Dataset	C2f module, EfficientRep, CIoU, DFL

c: PERFORMANCE METRICS AND SPEED

Benchmarked on the MS COCO dataset test-dev 2017, YOLOv8x achieved an impressive AP of 53.9% with an image size of 640 pixels, outperforming YOLOv5. YOLOv8 achieves a remarkable speed of 280 FPS on an NVIDIA A100 with TensorRT, emphasizing its efficiency in real-time applications.

With its advancements in architecture, loss functions, and segmentation capabilities, YOLOv8 stands as a powerful tool for a wide range of applications. Table 4 presents the key architectural innovations for the Yolo variants discussed in the preceding section.

Surveying the interconnections among the YOLO variants exposes a clear pattern, as shown in Table 4. The majority of YOLO variants reveal a connection to the Darknet framework, with later iterations progressing towards more sophisticated versions like CSPDarknet, as the underlying framework. Furthermore, Table 5 emphasizes a consistent trend, indicating that the COCO dataset serves as a key benchmark dataset for the majority of YOLO variants, comprising 80 object categories. These categories compromise of common objects like cars, bicycles, and animals, as well as more specific items such as umbrellas, handbags, and sports equipment, enabling a challenging benchmarking front. This shared framework and well recognised dataset benchmarking underscore the robust evolution and continuity in YOLOs development, providing a common foundation for testing and benchmarking across different architectural innovations.

IV. TRAINING STRATEGIES AND DATASET DIVERSITY

A. DATA AUGMENTATION TECHNIQUES

Within the fabric of YOLO's training strategy, data augmentation emerges as a dynamic and profound mechanism. The intricate orchestration of diverse transformations, including

TABLE 6. Comparative Study of YOLO variants.

Version	Date	Anchor	Framework	Backbone	MAP (%)	FPS	Detection Mechanism
YOLOv1	2015	×	Darknet	Darknet24	63.4	45	Single-stage
YOLOv2	2016	✓	Darknet	Darknet24	69.0	52	Single-stage
YOLOv3	2018	✓	Darknet	Darknet53	57.9	34	Single-stage
YOLOv4	2020	✓	Darknet	CSPDarknet53	44.3	65	Single-stage
YOLOv5	2020	✓	PyTorch	Modified CSP v7	50.7	200	Single-stage
YOLOv6	2022	×	PyTorch	EfficientRep	52.5	29	Single-stage
YOLOv7	2022	×	PyTorch	RepConvN	56.8	5-160	Single-stage
YOLOv8	2023	×	PyTorch	YOLO v8	53.9	280	Anchor-free

but not limited to random scaling, rotation, translation, illumination, and the popular Mosaic (YOLOv5) serves as a cornerstone for enhancing the variant robustness.

By exposing variants to a myriad of augmented instances during training, YOLO becomes adept at handling the inherent variations and complexities present in real-world scenarios. This augmentation strategy embebedded within the algorithmic pipeline not only mitigates the risk of overfitting but also fosters a model that generalizes effectively across diverse object appearances, orientations, and environmental conditions.

B. DYNAMIC TRAINING MECHANISMS

The training methodologies deployed across different variants of YOLO as presented in Table 5 underscore a continual evolution in optimizing object detection models. YOLOv1 initiated the journey with a grid-based approach leveraging the Darknet framework for training on the Pascal dataset.

Subsequent variants, like YOLOv2 and YOLOv3, expanded their horizons by incorporating hierarchical classification and adopting the Darknet-53 backbone, along with introducing innovative techniques such as the FPN.

YOLOv4 further enhanced the training process through techniques like enhanced quantization, PAN, and RepVGG. YOLOv5 marked a transition to PyTorch, embracing AutoAnchor, Mosaic and MixUp for improved performance.

YOLOv6 introduced advancements like RepVGG, PAN, and EfficientRep, while YOLOv7 continued to innovate with ELAN and model scaling. YOLOv8, developed in PyTorch, stands out with its C2f module, EfficientRep, CIoU, and DFL for robust and efficient training.

This iterative refinement in training techniques across YOLO versions showcases a commitment to optimizing object detection models through a diverse range of methodologies, each tailored to address the specific challenges and opportunities presented by evolving datasets.

V. YOLO VERSIONS: A COMPARATIVE ANALYSIS

This section provides a comparative analysis of the reviewed YOLO variants from YOLOv1 to YOLOv8, across a wide range of metrics, as presented in Table 6.

YOLOv1: Pioneer in Object Detection (2015) The inaugural version of YOLO, YOLOv1, introduced the groundbreaking concept of real-time object detection using a single-stage architecture with anchor boxes. Deploying the Darknet24 framework, it achieved a remarkable Mean Average Precision (mAP) of 63.4% while maintaining a processing speed of 45 frames per second (FPS).

YOLOv2: Refinements and Anchor Boxes (2016) Building upon the success of YOLOv1, YOLOv2 continued the utilization of anchor boxes for improved localization accuracy. Implemented within the Darknet24 framework, it achieved a notable increase in mAP, reaching 69.0%, and maintained real-time processing capabilities with 52 FPS.

YOLOv3: Multi-scale Features and Loss Functions (2018) YOLOv3 marked a balanced approach by adopting a multi-scale feature extraction architecture and introducing novel loss functions such as CIoU, GIoU, and BCE. Utilizing the Darknet53 framework, it achieved a mAP of 57.9% and demonstrated the ability to handle object detection across various scales at 34 FPS.

YOLOv4: Advanced Loss Functions (2020) With the adoption of the CSPDarknet53 framework, YOLOv4 emphasized advanced loss functions, including CIoU, DFL, and BCE, aiming to enhance bounding box accuracy while sustaining real-time processing. Despite a decrease in mAP to 44.3%, it exhibited a high FPS of 65.

YOLOv5: Leap in Accuracy and Efficiency (2020) A significant leap in accuracy and efficiency, YOLOv5, implemented the Modified CSP v7 architecture in PyTorch. With a single-stage detection mechanism and novel loss functions (CIoU, DFL, BCE), it achieved a mAP of 50.7% and a substantial increase in FPS to 200, showcasing its efficiency in real-time applications.

YOLOv6 to YOLOv8: Iterative Improvements (2022-2023) The subsequent iterations, YOLOv6, YOLOv7, and YOLOv8, demonstrate a commitment to iterative improvements. YOLOv6, utilizing the EfficientRep architecture, improved accuracy to 52.5%, while YOLOv7, based on the RepConvN, achieved a mAP of 56.8%. YOLOv8, introducing an anchor-free model, maintained a high accuracy of 53.9% with an impressive processing speed of 280 FPS.

TABLE 7. Diverse applications of YOLO.

Reference	Detection Type	Model	Key Characteristics	Performance Metrics
Pedestrain Detection				
	[58]	YOLO-R	Utilising YOLO-R, three Passthrough layers incorporating Route and Reorg layers were introduced to establish connections between pedestrian features in shallow and deep layers. The adjustment of the layer number in the Passthrough layer connection is a modification aimed at enhancing the network's capability to extract information from the shallow pedestrian features. The proposed method's performance was assessed through evaluation using the INRIA pedestrian dataset, achieving a processing speed of 25 frames per second.	25 FPS
	[59]	Multi-scale YOLO	Authors introduced a divide-and-rule method to address challenges, incorporating a segmentation function to split non-overlapping pedestrians into sub images, aiming to enhance detection. Multiresolution adaptive fusion was performed on the output of all images and sub images using a dedicated network architecture. The research extensively evaluates the proposed model on challenging pedestrian detection datasets, demonstrating its effectiveness in overcoming challenges and achieving improved detection performance.	91.6% MAP
Intrusion Detection				
	[60]	YOLO-v5	Based on YOLOv5, authors proposed a methodology for real-time detection, localisation, and recognition of actions from frames derived from a continuous video stream. The model analysed input frames at regular intervals, assigning an action label based on individual frames. The efficacy and speed of the YOLOv5 method in terms of recognition and localisation were showcased through experimentation with the Liris Human Activities dataset, validating the proposed approach's effectiveness.	0.19 FPS
Detection of Plant Disease				
	[61]	YOLO-v3	Authors focused on the application of YOLO-v3 for the timely and accurate detection of plant diseases. The research implemented lighter versions of YOLO known for their efficiency and high detection speed. A case study was conducted on the Papaya ringspot virus, a lethal viral disease impacting Papaya plants. To support the study, a dataset was constructed comprising images of both healthy and diseased leaves affected by the virus.	99.7 % MAP
	[26]	YOLO-v4	Authors deployed YOLO v4 based on the CSPDarknet53 framework and implemented channel pruning to streamline the detection component, enhancing overall efficiency. The fine-tuning process employed 2230 manually labelled apple flower images representing three varieties (Fuji, Red Love, and Gala). The proposed method demonstrated robustness in handling variations across different apple tree species and varying illumination conditions.	89.43% F1 score
	[62]	YOLO-v4	The study emphasised the importance of early pest detection for implementing effective crop defense strategies, based on the YOLOv4 architecture. To train the pest detection system, a custom dataset was collected in a realistic outdoor environment. The deployed system operated on an NVIDIA Jetson Xavier hardware platform, showcasing its practical application in real-world agricultural settings.	94.5% Precision

TABLE 7. *(Continued.)* **Diverse applications of YOLO.**

Human Facial Detection				
	[63]	YOLO-v3	Authors deployed YOLOv3 architecture for human face detection, demonstrating high detection speed tailored for real-time applications. It exhibited superior accuracy in detection and faster processing time for a 178x218 pixel image compared to similar target detection systems, highlighting its robustness and accelerated detection capabilities.	0.027 detection rate
	[64]	YOLO-v3	Addressed the challenge of facial recognition with varying scales. Incorporated anchor boxes more suited for facial features. Implemented a precise regression loss function achieving respectable accuracy.	78.3% Recall
Detecting Cancer				
	[65]	YOLO	Computer-Aided Diagnosis (CAD) system for mammograms. Utilised a ROI-based YOLO. Claiming to handle detection and classification simultaneously in one framework i.e., detecting and classifying masses in mammograms.	99.7% Accuracy
	[66]	YOLO-v1/v2/v3	Focused on optimising performance in automated detection of melanoma. Consideration of visual similarity between benign and malignant dermoscopic images. Need for fast and computationally effective systems for mobile applications targeting caregivers and homes.	77% MAP
Skin Disease Detection				
	[67]	YOLO-v3	Authors combined YOLO and GrabCut algorithms for skin lesion segmentation, aimed at addressing challenges in automatic segmentation of skin lesions. The proposed pipeline was based on: hair removal from image, lesion localisation, lesion area segmentation, and post-processing.	90% Recall
Pill Detection				
	[68]	YOLO-v3	Focused on safe administration of drugs to patients based on YOLOv3 detection architecture. Analysing performance of three models to determine the best pill recognition model (YOLO, SSD and RetinaNet).	82.89% Accuracy
Detection of Suspicious Activities				
	[69]	YOLO-v3	Addressing suspicious activity detection, authors utilised YOLOv3 with Darknet-53 as the feature extractor. Development pipeline consisted of video conversion into frames before undertaking activity examination. Key challenge was in human detection due to body flucuations whilst undertaking activities, overall performance was impressive.	96.42% F1 score
Facial and Mask Detection				
	[70]	YOLOv4	Focused on social distance and mask enforcement during the covid era, based on the YOLOv4 object detection architecture. Real-time inferencing of masks in video footage and images. Authors claim the selected architecture (YOLOv4) manifested faster inference speeds compared to other architectures with an overall fps of 38 on video data.	94.75% MAP
Traffic Detection				
	[71]	YOLO-v1	Focused on the detection of car license plates based on the YOLOv1 architecture based on the Darknet framework. YOLO's 7 convolutional layers were utilised for single-class detection and sliding-window mechanism for license plate detection. Training and validation was based on Taiwan's car license plates using an AOLP dataset.	98.22% Accuracy

TABLE 7. *(Continued.)* **Diverse applications of YOLO.**

	[72]	YOLO-v4	Focused on traffic sign detection for smart vehicles. Authors analysed YOLOv4 and YOLOv4-tiny combined with Spatial Pyramid Pooling (SPP), in addition to evaluating the impact of SPPP on feature extraction and learning object features. Evaluation metrics included MAP, working area size, detection time, and BFLOPS.	99.32% MAP
	[73]	YOLO-v4	WilDect-YOLO integrates a residual block into the CSPDarknet53 backbone, facilitating the extraction of robust and distinctive deep spatial features. To preserve essential feature information, the model incorporates DenseNet blocks, enhancing the overall retention of critical features. Additionally, the utilization of Spatial Pyramid Pooling (SPP) and a modified Path Aggregation Network (PANet) contributes to improved receptive field representation, preservation of fine-grain localized information, and enhanced feature fusion. The proposed model demonstrates remarkable performance metrics, achieving an mAP of 96.89% and an F1-score of 97.87%, all while maintaining a detection rate of 59.2 frames per second (FPS).	96.89% MAP
Robotic Application				
	[74]	YOLO-v2	YOLO-v2 is applied in a vision system designed for detecting stationary objects that could potentially obstruct the path of a mobile robot. The system employs the YOLO algorithm in conjunction with the Microsoft Kinect sensor to capture depth information, aiding in both object identification and distance calculation. To enhance the image processing algorithm's efficiency, the Nvidia Jetson TX2 GPU is utilized, resulting in faster and more effective object detection. Notably, the Microsoft camera exhibits an error rate below 3.64%.	3.64% (Error rate)
Drone-based Detection				
	[12]	YOLO-v3	Researchers have proposed the integration of the YOLO algorithm with UAV (Unmanned Aerial Vehicle) technology, forming YOLO-based UAV Technology (YBUT). The paper provides a comprehensive review of the practical applications of YBUT across diverse fields, including engineering, transportation, agriculture, and automation. Notably, the achieved mean Average Precision at 50% overlap (mAP50) is measured at 0.484, demonstrating the efficacy of YBUT in various real-world scenarios.	48.4% MAP
Garbage Detection				
	[75]	YOLO-v1	The paper addresses the pressing demand for effective garbage recycling solutions in response to the escalating volume of waste produced by human society. Existing machine learning models for garbage detection and classification face limitations, particularly in terms of slow processing speeds and large model sizes, rendering them impractical for real-time applications and portable edge-computing devices. The proposed model presents a fusion of the YOLO architecture with a Variational Autoencoder (VAE) to improve accuracy, processing speed, and model size in the context of garbage recycling.	60 FPS

TABLE 7. *(Continued.)* **Diverse applications of YOLO.**

PCB Component Detection	[76]	YOLO-v3	The accurate detection of electronic components on Printed Circuit Boards (PCBs) holds significant importance in ensuring quality control and intelligent assembly within the manufacturing processes of 3C (Computer, Communication, and Consumer Electronics) products. The paper proposes an enhancement to the YOLO model by scrutinizing the feature distribution of the Darknet-53 network's dimensionality-reduced output layers and the size distribution of detection targets. Based on this analysis, the original three YOLO output layers are adjusted to four, and a total of 12 anchor boxes are generated specifically for electronic component detection. This modification in the output layers of YOLO aims to optimize the model for the task at hand. Notably, the proposed approach achieves a Mean Average Precision (mAP) of 93.07%, underscoring its effectiveness in accurately identifying electronic components on PCBs.	93.07% MAP
Ship Detection	[77]	YOLO-v4	Authors moved away from conventional detection straegies to utilsing deep learning for ship detection based on YOLOv4. Authors leveraged multi-channel fusion to enhance the information available in SAR images and the feature extraction capabilities of the architecture.	90.30% AP
Pole Detection	[78]	YOLO-v3	Authors propose a mechanism for detection and counting poles in the distribution network based on UAV inspection, utilisng YOLOv3 as the detection architecture. Rationale for the selection of YOLOv3 was to leverage its high detection rates with a CNN utilised for detection of pole state.	90% Precision
Detecting Shuttlecock	[79]	YOLO-v2	Authors addressed the application of predicting the trajectory of a shuttlecock in two-dimensional space. The research was part of the development of a playing robot for shuttle badminton. The initial component focused on the detection of the shuttlecock using deep YOLOv2, with the authors reporting impressive performance.	96.3% Precision
Detecting Fires	[80]	YOLO-v4	This reserach looked into the application of a multi-scale detection mechanism for broadening the detection range and enhancing feature information via YOLOv4. Deployment of the Sequeeze and Excitation (SE) attention mechanism was undertaken for inter-channel feature fusion and enhanced feature extraction. Integration of picture transformation and migration learning was deployed to optimise convergence.	92.8% Precision
Defect System for Petrochemical Pipeline	[81]	YOLO-v5	Defect detection in petrochemical pipelines is paramount for mandating industrial production safety. With conventional approaches such as capturing CCTV videos of the pipeline's inner wall and human-led identification of defective areas, having many loopholes such as cost, time and bias, the authors propose an automated pipeline for detecting pipeline defects, based on YOLOv5.	92% F1-score

TABLE 7. *(Continued.)* **Diverse applications of YOLO.**

Wind Turbine Blades Detection	[82]	YOLO-v5	Authors proposed an SOD-YOLO model for surface defect detection, based on UAV image analysis and YOLOv5. The pipeline consisted of preprocessing steps, including foreground segmentation and applying Hough transform, to create the defect dataset. The authors proposed the integeration of a micro-scale detection layer to the original YOLOv5 architecture to improve the detection of small target defects.	95.1% MAP
Robot Detection	[83]	YOLO-v3	Authors aimed to enhance the detection precision (mAP) compared to standard methods. The proposed pipeline assumes the scene to be a plane with objects. The methodology is designed for use with autonomous robots. By leveraging the robot's dimensions and camera inclination angles, the spatial scale for each pixel in the input frame is predicted.	84% MAP
Signboard Detection	[84]	YOLO-v7	Authors experimented with the YOLOv7 architecture for Foreign Object Detection (FOD) and Wireless Power Transfer (WPT) systems.	97% MAP
Detecting Pothole	[85]	YOLO-v7	The research focused on the detection of road potholes aiming to mitigate car accidents and injuries. Conventional approaches for pothole detection involve the use of sensors, but this is considered expensive or practically infeasible. The proposed methodology utilises a smartphone's camera in addition to present location to detect and categorise potholes.	100% Precision
Vessel Model Detection	[86]	YOLO-v8	The research addressed marine pollution caused by debris entering ocean via rivers and aim to design effective control measures. A physical and bubble barrier system was developed to collect debris. Authors evaluated the effectiveness of deep learning architectures including YOLOv8 for real-time vessel detection and classification.	98.9% MAP

VI. REAL-WORLD APPLICATIONS AND IMPACT

A. SURVEILLANCE SYSTEMS AND PUBLIC SAFETY

YOLO's real-time processing capabilities make it invaluable in surveillance systems, enhancing public safety through the efficient monitoring of public spaces [46].

B. AUTONOMOUS VEHICLES AND TRAFFIC MANAGEMENT

In the realm of autonomous vehicles, YOLO plays a crucial role in object detection for obstacle avoidance and navigation [47]. Its rapid identification and classification of objects contribute to the safe and efficient operation of autonomous vehicles [48]. YOLO also supports traffic management systems by providing real-time information on road conditions [49].

C. INDUSTRIAL AUTOMATION AND QUALITY CONTROL

YOLO finds applications in industrial settings for automation and quality control [50]. In manufacturing, it can detect and inspect defects in products, ensuring adherence to quality standards [51]. The real-time nature of YOLO facilitates swift decision-making [52] in automated processes, contributing to increased efficiency [53] and reduced errors [54], in areas such as defect detection.

D. HEALTHCARE IMAGING AND DIAGNOSIS

In medical imaging, YOLO demonstrates efficacy in detecting and localizing abnormalities, aiding medical professionals in timely and accurate diagnoses in areas such as cancer and exudate detection for early diagnosis of deiabetic retinopathy [55]. YOLO's real-time processing is particularly

valuable in scenarios where quick decisions are critical for patient care.

E. ENVIRONMENTAL MONITORING AND WILDLIFE CONSERVATION

YOLO's adaptability extends to environmental monitoring, supporting wildlife conservation, biodiversity studies and renewable energy. It can detect and track animals in their natural habitats, aiding researchers in population monitoring and protection efforts. YOLO's real-time capabilities enhance the efficiency of conservation initiatives.

F. RETAIL AND CUSTOMER EXPERIENCE

In the retail domain, YOLO variants have been implemented to enhancing customer experiences and optimise several aspects of the supply chain. By leveraging its efficient object detection and tracking effeciency, YOLO variants can significantly contribute to automated inventory management, offering retailers real-time analysis of their stock levels and product availability [56] and [57].

To further illustrate YOLO's impact, Table 7 provides an overview of diverse applications and research studies leveraging YOLO. Each entry in the table highlights the reference, detection type, YOLO model used, key characteristics of the application, and performance metrics achieved. Notably, multiple applications have optimised the selected YOLO architecture for diverse purposes. While the majority of works presented, prioritised attaining high accuracy across metrics such as MAP, precision, and recall, certain works, driven by limitations in hardware resources or domain restrictions, directed efforts toward optimising Frames Per Second (FPS) for expedited inferencing, highlighting YOLOs versatility in adapting to the specific needs of different applications.

Another notable observation showcased in Table 7 is that most variants implemented are v3 onwards. This preference can be attributed to the crucial role played by YOLO-v3 as the initial variant addressing the challenge of small object detection. YOLO-v3 introduced multi-scale detection mechanisms with subsequent variants i.e., PANet (YOLOv4), building on this concept, thereby unlocking applicability in scenarios where the detection of small targets was essential.

VII. CHALLENGES

Despite its remarkable success, YOLO faces certain challenges and areas for improvement. This section critically examines the limitations of the YOLO framework, proposes potential avenues for future research to address these challenges, and explores the integration of YOLO with edge deployment and federated learning for enhanced privacy and adaptability:

A. HANDLING OCCLUSIONS AND CLUTTER

One persistent challenge for YOLO is effectively handling occluded objects and scenes with high clutter. In scenarios where objects overlap or are partially obscured [87], YOLO may struggle to accurately detect and delineate individual instances. Future research could explore novel approaches, such as improved feature representations or context-aware models, to enhance YOLO's ability to cope with occlusions and cluttered scenes [88].

B. SCALE VARIATIONS AND FINE-GRAINED OBJECT DETECTION

The robust detection of objects at varying scales and the identification of fine-grained details remain areas where YOLO can be refined [89]. Adapting the architecture to better handle small or distant objects, potentially through multi-scale feature fusion strategies, could elevate YOLO's performance in scenarios demanding fine-grained object detection. The integration of federated learning can contribute to the enhancement of YOLO's adaptability across diverse scales by leveraging collaborative learning from edge devices.

C. DOMAIN ADAPTATION AND GENERALIZATION

While YOLO has showcased versatility across domains, there is room for improvement in domain adaptation [90]. Ensuring robust performance when transitioning from one environment to another [4], especially in scenarios with significant domain shifts, is a challenge. The integration of federated learning introduces a collaborative approach to domain adaptation, allowing YOLO models to adapt to diverse edge environments through decentralized learning.

D. EXPLAINABILITY AND INTERPRETABILITY

As with any machine learning system, addressing biases in training data and ensuring ethical considerations are paramount [91]. YOLO, like other object detection models, may exhibit biases that mirror the biases present in the data it was trained on. The integration of federated [92] learning can contribute to addressing biases by ensuring a more diverse and representative dataset across edge devices, enhancing the fairness and interpretability [93]of YOLO models.

E. ADDRESSING BIASES AND ETHICAL CONSIDERATIONS

As YOLO evolves, considerations for privacy preservation become increasingly important. The integration of YOLO with federated learning aligns with privacy-preserving objectives by allowing models to be trained collaboratively across edge devices without centralizing sensitive data [94]. This integration addresses ethical considerations related to data privacy in various applications, from surveillance to healthcare.

F. REAL-TIME PROCESSING OPTIMIZATION

While YOLO is renowned for its real-time processing capabilities, continuous optimization in this aspect is essential [95]. Future research may explore innovative techniques for further improving inference speed without compromising accuracy. The integration of edge deployment and federated

learning introduces a decentralized approach to real-time processing, where models are trained collaboratively on edge devices, contributing to enhanced efficiency.

G. EDGE DEPLOYMENT AND FEDERATED LEARNING

The deployment of YOLO at the edge and the integration with federated learning present exciting opportunities [96]. Edge devices benefit from YOLO's efficiency, enabling on-device object detection without relying heavily on centralized servers [97]. Federated learning introduces a collaborative training paradigm where YOLO models are trained across multiple edge devices [98], enhancing privacy, adaptability, and generalization [99]. This integration aligns with the evolving landscape of decentralized and privacy-preserving machine learning.

VIII. CONCLUSION

As we conclude this comprehensive exploration of YOLOs evolution, challenges, and integrations, it becomes evident that YOLO has not only shaped the landscape of object detection but continues to evolve dynamically, staying at the forefront of advancements in computer vision. From the pioneering YOLOv1 to the sophisticated YOLOv8, the architectural innovations and training strategies have propelled YOLO into the limelight, making it a go-to choice for real-time object detection.

Reviewing the first objective of this review, it is evident that YOLO variants have endured significant architectural innovations during evolution. This progression includes highlights such as the introduction of Feature Pyramid Networks (FPN) in YOLOv3 and the incorporation of ELAN mechanisms in YOLOv7. Notably, the later variants have acknowledged the requirement for versatility to meet the diverse demands of industrial deployments. To address this, researchers have proposed several sub-variants of each architecture, such as v5s/m/l/x, each with varying internal architectural configurations. This approach enables developers to select a base architecture based on their specific requirements for accuracy and detection rate requirements. The resulting versatility has permitted YOLO variants to successfully penetrate various applications in the industry as evident from Table 7.

The second objective of the paper, which scrutinizes the training strategies for performance optimisation, reveals a comprehensive analysis of training methodologies across YOLO variants. As presented in Table 5, each variant not only endured testing on key benchmark datasets but also engaged in in-depth tuning of internal architectures. YOLOv4, for instance, transitioned from Darknet53 to CSPDarknet53, demonstrating a shift in architectural choices for enhanced performance. In the case of YOLOv6, the focus moved towards training optimisation through EfficientRep, followed by RepConvN (YOLOv7), indicating a deliberate effort to incorporate incremental training boosters.

These refined training strategies have bestowed developers with a rich selection pool, enabling them to select methodologies based on their specific domain requirements. This diversity is evident in the extensive range of domains presented in the third objective, highlighting YOLO variant deployments across various industries. The incremental advances in training strategies contribute significantly to the adaptability and performance optimization of YOLO variants in real-world applications.

In considering future challenges, it is envisioned that YOLO variants will continue to address and improve performance on small object targets, especially as they penetrate into more specialized areas such as precision manufacturing. This trajectory suggests a necessity for advancements in lightweight architectures that balance high accuracy with stringent FPS requirements. As YOLO progresses, meeting the demands of niche applications will likely drive further innovation in architectural design and optimisation, ensuring its continued relevance in domains with stringent requirements for precision and efficiency.

REFERENCES

[1] M. Hussain and R. Hill, "Custom lightweight convolutional neural network architecture for automated detection of damaged pallet racking in warehousing & distribution centers," *IEEE Access*, vol. 11, pp. 58879–58889, 2023.

[2] M. Hussain, "YOLO-v5 variant selection algorithm coupled with representative augmentations for modelling production-based variance in automated lightweight pallet racking inspection," *Big Data Cognit. Comput.*, vol. 7, no. 2, p. 120, Jun. 2023.

[3] M. F. Talu, K. Hanbay, and M. H. Varjovi, "CNN-based fabric defect detection system on loom fabric inspection," *Tekstil Konfeksiyon*, vol. 32, no. 3, pp. 208–219, Sep. 2022.

[4] B. A. Aydin, M. Hussain, R. Hill, and H. Al-Aqrabi, "Domain modelling for a lightweight convolutional network focused on automated exudate detection in retinal fundus images," in *Proc. 9th Int. Conf. Inf. Technol. Trends (ITT)*, May 2023, pp. 145–150.

[5] M. A. Ansari, A. Crampton, and S. Parkinson, "A layer-wise surface deformation defect detection by convolutional neural networks in laser powder-bed fusion images," *Materials*, vol. 15, no. 20, p. 7166, Oct. 2022.

[6] P. Lala Mehta and A. Kumar, "Livai: A novel resource-efficient real-time facial emotion recognition system based on a custom deep CNN model," *SSRN Electron. J.*, Feb. 2022.

[7] 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.

[8] A. Koubaa, A. Ammar, A. Kanhouch, and Y. AlHabashi, "Cloud versus edge deployment strategies of real-time face recognition inference," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 1, pp. 143–160, Jan. 2022.

[9] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proc. IEEE*, vol. 111, no. 3, pp. 257–276, Mar. 2023.

[10] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of YOLO algorithm developments," *Proc. Comput. Sci.*, vol. 199, pp. 1066–1073, Jan. 2022.

[11] P. P. Khaire, R. D. Shelke, D. Hiran, and M. Patil, "Comparative study of a computer vision technique for locating instances of objects in images using YOLO versions: A review," in *Proc. Int. Conf. Inf. Commun. Technol. Intell. Syst.*, Springer, 2023, pp. 349–359.

[12] C. Chen, Z. Zheng, T. Xu, S. Guo, S. Feng, W. Yao, and Y. Lan, "YOLO-based UAV technology: A review of the research and its applications," *Drones*, vol. 7, no. 3, p. 190, Mar. 2023.

[13] X. Qian, B. Wu, G. Cheng, X. Yao, W. Wang, and J. Han, "Building a bridge of bounding box regression between oriented and horizontal object detection in remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023.

[14] X. Qian, Y. Huo, G. Cheng, C. Gao, X. Yao, and W. Wang, "Mining high-quality pseudoinstance soft labels for weakly supervised object detection in remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023.

[15] L. Li, X. Yao, X. Wang, D. Hong, G. Cheng, and J. Han, "Robust few-shot aerial image object detection via unbiased proposals filtration," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023.
[16] S. Agarwal, J. O. D. Terrail, and F. Jurie, "Recent advances in object detection in the age of deep convolutional neural networks," 2019, *arXiv:1809.03193*.
[17] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," 2018, *arXiv:1809.02165*.
[18] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," 2020, *arXiv:1911.11929*.
[19] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3500–3509.
[20] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
[22] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.
[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[24] C. Sun, Y. Ai, S. Wang, and W. Zhang, "Dense-RefineDet for traffic sign detection and classification," *Sensors*, vol. 20, no. 22, p. 6570, Nov. 2020.
[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2017, *arXiv:1708.02002*.
[26] D. Wu, S. Lv, M. Jiang, and H. Song, "Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments," *Comput. Electron. Agricult.*, vol. 178, Nov. 2020, Art. no. 105742.
[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[29] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.
[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[31] J. Redmon and A. Ali, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 7263–7271.
[32] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," 2014, *arXiv:1405.0312*.
[33] J. Redmon and A. Ali, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[34] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[36] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.
[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[38] Z. Ma, M. Li, and Y. Wang, "PAN: Path integral based convolution for deep graph neural networks," 2019, *arXiv:1904.10996*.
[39] G. Jocher et al., (2020), "ultralytics/yolov5:v3.0," *Zenodo*, doi: 10.5281/zenodo.3983579.
[40] Z. Yao, Y. Cao, S. Zheng, G. Huang, and S. Lin, "Cross-iteration batch normalization," 2021, *arXiv:2002.05712*.
[41] C.-Y. Wang, A. Bochkovskiy, and H.-Y. Liao. (2022). *YOLOv6*. GitHub. [Online]. Available: https://github.com/meituan/YOLOv6
[42] 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*.
[43] C.-Y. Wang, A. Bochkovskiy, and H.-Y. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022, *arXiv:2207.02696*.
[44] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "RepVGG: Making VGG-style ConvNets great again," 2021, *arXiv:2101.03697*.
[45] J. Solawetz, "What is YOLOv8? The ultimate guide," Tech. Rep., Jan. 2023.
[46] D. Beymer, "Person counting using stereo," in *Proc. Workshop Human Motion*, Dec. 2000, pp. 127–133.
[47] M. Nagy and G. Lăzăroiu, "Computer vision algorithms, remote sensing data fusion techniques, and mapping and navigation tools in the Industry 4.0-based Slovak automotive sector," *Mathematics*, vol. 10, no. 19, p. 3543, Sep. 2022.
[48] S. Battiato, S. Conoci, R. Leotta, A. Ortis, F. Rundo, and F. Trenta, "Benchmarking of computer vision algorithms for driver monitoring on automotive-grade devices," in *Proc. AEIT Int. Conf. Electr. Electron. Technol. Automot. (AEIT AUTOMOTIVE)*, Nov. 2020, pp. 1–6.
[49] J. Barthélemy, N. Verstaevel, H. Forehead, and P. Perez, "Edge-computing video analytics for real-time traffic monitoring in a smart city," *Sensors*, vol. 19, no. 9, p. 2048, May 2019.
[50] L. Scime and J. Beuth, "Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm," *Additive Manuf.*, vol. 19, pp. 114–126, Jan. 2018.
[51] N. Lyons, "Deep learning-based computer vision algorithms, immersive analytics and simulation software, and virtual reality modeling tools in digital twin-driven smart manufacturing," *Econ., Manage., Financial Markets*, vol. 17, no. 2, pp. 67–81, 2022.
[52] K. Li, E. D. Miller, M. Chen, T. Kanade, L. E. Weiss, and P. G. Campbell, "Computer vision tracking of stemness," in *Proc. 5th IEEE Int. Symp. Biomed. Imag., Nano Macro*, May 2008, pp. 847–850.
[53] Q.-J. Zhao, P. Cao, and D.-W. Tu, "Toward intelligent manufacturing: Label characters marking and recognition method for steel products with machine vision," *Adv. Manuf.*, vol. 2, no. 1, pp. 3–12, Mar. 2014.
[54] S. Paneru and I. Jeelani, "Computer vision applications in construction: Current state, opportunities & challenges," *Autom. Construct.*, vol. 132, Dec. 2021, Art. no. 103940.
[55] M. Hussain, H. Al-Aqrabi, M. Munawar, R. Hill, and S. Parkinson, "Exudate regeneration for automated exudate detection in retinal fundus images," *IEEE Access*, vol. 11, pp. 83934–83945, 2022.
[56] P. Cortez, L. M. Matos, P. J. Pereira, N. Santos, and D. Duque, "Forecasting store foot traffic using facial recognition, time series and support vector machines," in *Proc. Int. Joint Conf.* Cham, Switzerland: Springer, 2017, pp. 267–276.
[57] N. James, "Automated checkout for stores: A computer vision approach," *Revista Gestão Inovação Tecnologias*, vol. 11, no. 3, pp. 1830–1841, Jun. 2021.
[58] W. Lan, J. Dang, Y. Wang, and S. Wang, "Pedestrian detection based on YOLO network model," in *Proc. IEEE Int. Conf. Mechatronics Autom. (ICMA)*, Aug. 2018, pp. 1547–1551.
[59] W.-Y. Hsu and W.-Y. Lin, "Adaptive fusion of multi-scale YOLO for pedestrian detection," *IEEE Access*, vol. 9, pp. 110063–110073, 2021.
[60] S. Shinde, A. Kothari, and V. Gupta, "YOLO based human action recognition and localization," *Proc. Comput. Sci.*, vol. 133, pp. 831–838, Jan. 2018.
[61] P. Maski and A. Thondiyath, "Plant disease detection using advanced deep learning algorithms: A case study of papaya ring spot disease," in *Proc. 6th Int. Conf. Image, Vis. Comput. (ICIVC)*, Jul. 2021, pp. 49–54.
[62] M. Lippi, N. Bonucci, R. F. Carpio, M. Contarini, S. Speranza, and A. Gasparri, "A YOLO-based pest detection system for precision agriculture," in *Proc. 29th Medit. Conf. Control Autom. (MED)*, Jun. 2021, pp. 342–347.
[63] W. Yang and Z. Jiachun, "Real-time face detection based on YOLO," in *Proc. 1st IEEE Int. Conf. Knowl. Innov. Invention (ICKII)*, Jul. 2018, pp. 221–224.
[64] W. Chen, H. Huang, S. Peng, C. Zhou, and C. Zhang, "YOLO-Face: A real-time face detector," *Vis. Comput.*, vol. 37, no. 4, pp. 805–813, Mar. 2020.

[65] M. A. Al-masni, M. A. Al-antari, J.-M. Park, G. Gi, T.-Y. Kim, P. Rivera, E. Valarezo, M.-T. Choi, S.-M. Han, and T.-S. Kim, "Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system," *Comput. Methods Programs Biomed.*, vol. 157, pp. 85–94, Apr. 2018.

[66] Y. Nie, P. Sommella, M. O'Nils, C. Liguori, and J. Lundgren, "Automatic detection of melanoma with YOLO deep convolutional neural networks," in *Proc. E-Health Bioeng. Conf. (EHB)*, Nov. 2019, pp. 1–4.

[67] H. M. Ünver and E. Ayan, "Skin lesion segmentation in dermoscopic images with combination of YOLO and GrabCut algorithm," *Diagnostics*, vol. 9, no. 3, p. 72, Jul. 2019.

[68] L. Tan, T. Huangfu, L. Wu, and W. Chen, "Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification," *BMC Med. Informat. Decis. Making*, vol. 21, no. 1, Nov. 2021.

[69] N. Bordoloi, A. K. Talukdar, and K. K. Sarma, "Suspicious activity detection from videos using YOLOv3," in *Proc. IEEE 17th India Council Int. Conf. (INDICON)*, Dec. 2020, pp. 1–5.

[70] K. Bhambani, T. Jain, and K. A. Sultanpure, "Real-time face mask and social distancing violation detection system using YOLO," in *Proc. IEEE Bengaluru Humanitarian Technol. Conf. (B-HTC)*, Oct. 2020, pp. 1–6.

[71] Hendry and R.-C. Chen, "Automatic license plate recognition via sliding-window darknet-YOLO deep learning," *Image Vis. Comput.*, vol. 87, pp. 47–56, Jul. 2019.

[72] C. Dewi, R.-C. Chen, X. Jiang, and H. Yu, "Deep convolutional neural network for enhancing traffic sign recognition developed on YOLO v4," *Multimedia Tools Appl.*, vol. 81, no. 26, pp. 37821–37845, Apr. 2022.

[73] A. M. Roy, J. Bhaduri, T. Kumar, and K. Raj, "WilDect-YOLO: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection," *Ecolog. Informat.*, vol. 75, Jul. 2023, Art. no. 101919.

[74] D. H. Dos Reis, D. Welfer, M. A. D. S. L. Cuadros, and D. F. T. Gamarra, "Mobile robot navigation using an object recognition software with RGBD images and the YOLO algorithm," *Appl. Artif. Intell.*, vol. 33, no. 14, pp. 1290–1305, Nov. 2019.

[75] A. Ye, B. Pang, Y. Jin, and J. Cui, "A YOLO-based neural network with VAE for intelligent garbage detection and classification," in *Proc. 3rd Int. Conf. Algorithms, Comput. Artif. Intell.*, Dec. 2020.

[76] J. Li, J. Gu, Z. Huang, and J. Wen, "Application research of improved YOLO v3 algorithm in PCB electronic component detection," *Appl. Sci.*, vol. 9, no. 18, p. 3750, Sep. 2019.

[77] J. Jiang, X. Fu, R. Qin, X. Wang, and Z. Ma, "High-speed lightweight ship detection algorithm based on YOLO-V4 for three-channels RGB SAR image," *Remote Sens.*, vol. 13, no. 10, p. 1909, May 2021.

[78] B. Chen and X. Miao, "Distribution line pole detection and counting based on YOLO using UAV inspection line video," *J. Electr. Eng. Technol.*, vol. 15, no. 1, pp. 441–448, Jul. 2019.

[79] S. R. Vrajesh, A. N. Amudhan, A. Lijiya, and A. P. Sudheer, "Shuttlecock detection and fall point prediction using neural networks," in *Proc. Int. Conf. Emerg. Technol. (INCET)*, Jun. 2020, pp. 1–6.

[80] H. Wu, Y. Hu, W. Wang, X. Mei, and J. Xian, "Ship fire detection based on an improved YOLO algorithm with a lightweight convolutional neural network model," *Sensors*, vol. 22, no. 19, p. 7420, Sep. 2022.

[81] K. Chen, H. Li, C. Li, X. Zhao, S. Wu, Y. Duan, and J. Wang, "An automatic defect detection system for petrochemical pipeline based on cycle-GAN and YOLO v5," *Sensors*, vol. 22, no. 20, p. 7907, Oct. 2022.

[82] R. Zhang and C. Wen, "SOD-YOLO: A small target defect detection algorithm for wind turbine blades based on improved YOLOv5," *Adv. Theory Simulations*, vol. 5, no. 7, Jul. 2022, Art. no. 2100631.

[83] I. Khokhlov, E. Davydenko, I. Osokin, I. Ryakin, A. Babaev, V. Litvinenko, and R. Gorbachev, "Tiny-YOLO object detection supplemented with geometrical data," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, May 2020, pp. 1–5.

[84] Y. A. Khan, S. Imaduddin, A. Ahmad, and Y. Rafat, "Image-based foreign object detection using YOLO v7 algorithm for electric vehicle wireless charging applications," in *Proc. 5th Int. Conf. Power, Control Embedded Syst. (ICPCES)*, Jan. 2023, pp. 1–6.

[85] E. S. T. K. Reddy and V. Rajaram, "Pothole detection using CNN and YOLO v7 algorithm," in *Proc. 6th Int. Conf. Electron., Commun. Aerosp. Technol.*, Dec. 2022, pp. 1255–1260.

[86] A. Munin, A. Folarin, A. Munin-Doce, L. Alonso-Garcia, V. Diaz-Casas, S. Ferreno-Gonzalez, and J. M. Ciriano-Palacios, "Real time vessel detection model using deep learning algorithms for controlling a barrier system," *J. SSRN*, Apr. 2023.

[87] M. Ghafoor and A. Mahmood, "Quantification of occlusion handling capability of 3D human pose estimation framework," *IEEE Trans. Multimedia*, 2022.

[88] M. F. Aslan, A. Durdu, K. Sabanci, and M. A. Mutluer, "CNN and HOG based comparison study for complete occlusion handling in human tracking," *Measurement*, vol. 158, Jul. 2020, Art. no. 107704.

[89] H. T. Mustafa, J. Yang, and M. Zareapoor, "Multi-scale convolutional neural network for multi-focus image fusion," *Image Vis. Comput.*, vol. 85, pp. 26–35, May 2019.

[90] A. Zahid, M. Hussain, R. Hill, and H. Al-Aqrabi, "Lightweight convolutional network for automated photovoltaic defect detection," in *Proc. 9th Int. Conf. Inf. Technol. Trends (ITT)*, May 2023, pp. 133–138.

[91] D. S. Char, N. H. Shah, and D. Magnus, "Implementing machine learning in health care—Addressing ethical challenges," *New England J. Med.*, vol. 378, no. 11, pp. 981–983, Mar. 2018.

[92] A. Lakhan, M. A. Mohammed, K. H. Abdulkareem, H. Hamouda, and S. Alyahya, "Autism spectrum disorder detection framework for children based on federated learning integrated CNN-LSTM," *Comput. Biol. Med.*, vol. 166, Nov. 2023, Art. no. 107539.

[93] H. Younes, H. L. Blevec, M. Léonardon, and V. Gripon, "Interoperability of compression techniques for efficient deployment of CNNs on microcontrollers," in *Proc. Int. Conf. Syst.-Integr. Intell.*, Springer, 2022, pp. 543–552.

[94] N. Rane, S. Choudhary, and J. Rane, "YOLO and faster R-CNN object detection in architecture, engineering and construction (AEC): Applications, challenges, and future prospects," *Eng. Construction, Appl., Challenges, Future Prospects*, Oct. 2023.

[95] B.-G. Han, J.-G. Lee, K.-T. Lim, and D.-H. Choi, "Design of a scalable and fast YOLO for edge-computing devices," *Sensors*, vol. 20, no. 23, p. 6779, Nov. 2020.

[96] G. Plastiras, M. Terzi, C. Kyrkou, and T. Theocharidcs, "Edge intelligence: Challenges and opportunities of near-sensor machine learning applications," in *Proc. IEEE 29th Int. Conf. Application-specific Syst., Architectures Processors (ASAP)*, Jul. 2018, pp. 1–7.

[97] M. P. Véstias, "A survey of convolutional neural networks on edge with reconfigurable computing," *Algorithms*, vol. 12, no. 8, p. 154, Jul. 2019.

[98] Q. Wang, Q. Li, K. Wang, H. Wang, and P. Zeng, "Efficient federated learning for fault diagnosis in industrial cloud-edge computing," *Computing*, vol. 103, no. 10, pp. 2319–2337, Oct. 2021.

[99] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large CNNs at the edge," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 14068–14080.

MUHAMMAD HUSSAIN received the B.Eng. degree in electrical and electronic engineering and the M.S. degree in Internet of Things from the University of Huddersfield, in 2019, and the Ph.D. degree in artificial intelligence for defect identification. He is an accomplished Researcher hailing in Dewsbury, U.K. His work contributes to optimizing PV systems' efficiency and reliability. He is equally passionate about machine vision, focusing on lightweight architectures for edge device deployment in real-world production settings. Beyond fault detection, he explores AI interpretability, concentrating on developing explainable AI for medical and healthcare applications. His interdisciplinary approach underscores his commitment to ethical and impactful AI solutions. With his diverse expertise spanning AI, fault detection, machine vision, and interpretability, he aims to leave his mark on shaping the future of technology and its positive influence on society. His research interests include fault detection, particularly microcracks on photovoltaic (PV) cells due to mechanical and thermal stress.