Object Detection using Deep-Learning Techniques: A Comparative Study

Sharjeel Hussain[1], Samina Khalid[2], Afraz Hussain Majeed[3], Sajjad Manzoor[1,*]

[1]Department of Electrical Engineering, Mirpur University of Science and Technology, Mirpur AJK, Pakistan
[2]Department of Computer Science and Information Technology, Mirpur University of Science and Technology, Mirpur AJK, Pakistan
[3]School of Energy and Power Engineering, Jiangsu University, Zhenjiang, China
[*]Corresponding author: sajjad.ee@must.edu.pk

Abstract

These days, object detection is one of the important research problems in computer vision, used in applications such as real-time surveillance, security, self-driven vehicles, robotics, human-computer interaction, and image retrieval. Where accurate classification and localization of objects are performed for these applications. This can be achieved through deep learning-based detection techniques, one of the most widely used contemporary approaches, since they have high success rates. This paper presents a comprehensive review of recent advancements in deep learning-based object detection, focusing on notable algorithms such as Faster R-CNN, SSD, YOLOv4, and YOLOv5. In addition, we have also investigated the latest advances in the YOLO family, including YOLOv7 and YOLOv8, which have brought architectural improvements for improvement in detection speed and accuracy. These detection parameters are verified by implementing and testing these algorithms on similar datasets for a comparative analysis of their performance. Detection experiments are conducted using GPU hardware acceleration, with the primary objectives being real-time detection and minimizing error rates. The comparative results provide valuable information about the strengths and weaknesses of each algorithm for real-world applications.

Keywords—Deep learning-based object detection: YOLO, Single-shot detection, Bounding boxes

1 Introduction

Object detection is one of the most compelling techniques used in recent days to recognize and locate desired objects from a digital image or video. Its applications mostly include real-time surveillance, autonomous driving, robot vision, security, human-aided machine interface, and image recovery. These tasks are accomplished with the help of various measurement sensors and algorithms. Object detection can be used to recognize and locate desired objects from digital images or videos. The object to be detected may be a human being, a vehicle, or an animal. Object detection is a basis for many applications related to face recognition, character recognition, and video analysis [1]. The object detection process can be performed for one of two reasons; firstly, it can be used to detect the existence of a particular object, such as a specific building, a wanted person illegal vehicle, etc. Secondly, to detect some predefined categories related to objects, e.g., plants, humans, cats, dogs, mobile phones, books, etc.

Different challenges regarding visual recognition are object instance segmentation [2], image classification [3], semantic segmentation [4], and detection [5]. In the instance segmentation, different objects are identified by their individual pixel-level mask called "pixel-level localization. On the other hand, in semantic segmentation, a précised category label is assigned to every pixel-by-pixel classifier in order to extract more detailed information from an image [6]. If only semantic segmentation of the object is desired, the image classification technique is used. While in object detection, recognition, and location of objects in an image are defined. This is done by putting a bounding box around the targeted object.

Initially, object detection used proposal generation, feature vector extraction, and region classification. In proposal generation, the aim for desired objects is achieved in a given image by generating a "region of interest" (ROI). The image is scanned by sliding

ISSN: 2523-0379 (Online), ISSN: 1605-8607 (Print)
DOI: https://doi.org/10.52584/QRJ.2202.04

windows to extract data. In the second step, in order to extract semantic data from roi, a fixed-length feature vector obtained from the sliding window is used. This feature vector is encoded by Scale Invariant Feature Transform (SIFT) [7], Haar [8], Histogram of Gradients (HOG) [9], or Speeded Up Robust Features (SURF) [10]. In the last step, labels are assigned to objects according to their categories. For region classification, support vector machines (SVM) are commonly used, due to their better performance in small-scale training data [11].

The traditional methods use feature representation techniques to generate regions of interest. It is observed that with the help of better feature representation and precise regional classification, notable performance can be obtained [12]. With respect to generic feature extraction, the Hough transform, while for corner feature extraction, Harris corner detection can be used. Comer features are obtained from two images using the method, and the correlation degree between their points is calculated to detect objects. However, the traditional methods showed limitations during proposal generation, feature description, and global solution [13]. A large number of unnecessary proposals were generated during the first step. This creates confusion during classification. In complex scenarios, the manually designed window scaling is unable to describe objects. Moreover, since each detection step is designed and optimized individually, the global solution is not fully optimized. Deep convolutional networks have overtaken the limitations of traditional methods and have accomplished notable success in object detection.

Neural networks are biologically inspired networks [14] first proposed for image recognition. However, the early approach does not have a better optimization algorithm for supervised learning. This research gap was filled later on by researchers over time. An optimized convolutional network with stochastic gradient descent shows better performance on digital recognition. Although deep neural networks also have some limitations, the enormous out-of-order training data can cause misleading results. Similarly, limited computational sources and poor theoretical sustenance also need to be addressed [15]. Object detection techniques based on deep learning are highly popular for unusual-level vision tasks used for real-time object detection [16]. Furthermore, along with the development of GPU computing power, deep networks bring evolution in object detection [17]. In this paper, we will address major efforts in deep learning-based object detection. A comparison, that is based on individual performance of different algorithms, including Faster RCNN, SSD, YOLOv4, and YOLOv5 are will be submitted. Real-time object detection and a single sample image will be used. For comparison, both speed and accuracy will be taken into consideration.

The paper is further organized as follows: in Section 2, the development and work related to Faster RCNN, SSD, YOLOv4, and YOLOv5 are given. The architectures of deep learning algorithms, i.e., Faster RCNN, SSD, YOLOv4, and YOLOv5, are given in Section 3. The method to compare various algorithms for object detection in an image or video is also given in this section. The implementation and comparison of various algorithms for object detection are given in Section 4. The paper is concluded in Section 5.

2 Literature Review

In this paper, a comprehensive survey of new advances regarding deep learning-based object detection is given. Moreover, four algorithms, RCNN, SSD, YOLOV4, and YOLOV5, are implemented. Their performance is tested on similar input data. Deep learning-based object detection is achieved by two main methods. One is the single-stage method, where object localization and object classification are achieved by a single network; examples are YOLO and SSD [18]. Another is the Double-stage method, where two separated networks are used for localization and classification, such as Faster R-CNN. The state-of-the-art means applying deep learning networks as pillars, while detection networks are used to identify features from input data [19]. For accurate localization and precise object detection, double-stage networks are used, while a single-stage network is used for fast bounding box prediction. In double-stage detectors, at the first stage, the network that predicts bounding boxes is called the region proposal network. The second stage is classification, in which features of an object are extracted, called bounding box regression. The single-stage detector directly predicts bounding boxes without the help of a region proposal network, making it time-efficient and more suitable for real-time applications [20].

With recent advancements in data science, deep neural networks have solved problems of big data analytics with variations in size and properties [21]. DNNs have been proven powerful tools for computer vision since they perform a number of standard tasks, including detecting moving objects in real-time along with background scenes, shadows, and light intensity variation [22]. In [23] comparison of object detection using deep learning with state-of-the-art methods is done to find out the robustness of improved algorithms

using Background Challenges Model (BCM) datasets, while a survey on real-time multi-object detection with deep learning approaches is presented in [24]. Moreover, the use of deep learning models called multi-object tracking (MOT) techniques to handle the challenges of computer vision to track the number of moving objects is given. In [25], a comparison of Faster Region-based Convolutional Neural Networks (Faster R-CNN), SSD, and YOLO has been given to discover their performance on image detection. Furthermore, the Microsoft COCO dataset is used to analyze strengths and weaknesses on the basis of some parameters, i.e., accuracy and precision. Object detection procedure depends upon two components or steps [26]-[27]. In object detection, the first step to be considered is detection pattern settings, i.e., box-level and pixel/mask-level techniques. Another consideration is to differentiate between algorithms it is a stage detector or a stage detector.

There are two types of settings for localization: bounding box-level localization and Mask or pixel-level localization [28]. In the first type of detection algorithm, box interpretation is required, and performance is measured by calculating the Intersection over Union (IoU) of the predicted box and ground truth box. Another setting, called instance segmentation, is based on pixel or mask level and used to segment each object as compared to the roughly generated bounding box. Deep learning-based state-of-the-art algorithms are classified into two types [29]; i.e., one-stage detectors and two-stage-detectors. One-stage-detectors have a single shot for proposal generation plus image classification [30]. However, these single-step algorithms are faster but are not accurate state-of-the-art methods. Two-stage detectors complete the detection process in two steps [31]-[36]. In the first step, a set of proposals is generated then in the second step deep learning network will be used to encode the features of the generated set of proposals to predict the class of objects [37]-[39]. In the first step, deep neural learning networks will be able to find the proposed region of images or objects. While feature extraction, classification, and making predictions of objects [40] are done in the second step, where DNN-based models are used to categorize or classify with exact labeling. Moreover, a region is basically a given set of objects or backgrounds that is refined by the proposed model. It has typically state-of-the-art accuracy and therefore has a slower processing time.

The deep-learning-based object detection is therefore achieved by two main methods. YOLO and SSD are single-stage methods; object localization and object classification are achieved in a single step [41]-[46]. In Faster R-CNN two-stage method is used, which utilizes two separate networks for localization and classification [47]-[49]. The state-of-the-art means applying deep learning networks as pillars, while detection networks are used to identify features from input data [50]. For accurate localization and precise object detection, double-stage networks are used, while a single-stage network is used for fast bounding box prediction. In double-stage detectors, at the first stage, the network that predicts bounding boxes is called the region proposal network. The second stage is classification, in which features of an object are extracted, called bounding box regression. The single-stage detector directly predicts bounding boxes without the help of a region proposal network, making it time-efficient and more suitable for real-time applications [51].

You Only Look Once (YOLO), the initial form available in 2016, called YOLOv1, is a single-stage real-time detection algorithm [52]-[57], where an image is divided spatially into a 7x7 grid. In an ideal implementation, each cell presents one or more objects; however, in a real case, two adjacent centers of objects are considered. In [58], they used YOLO to solve real-life difficulties for diverse smart city applications, such as parking occupancy detection. The YOLO architecture has a simple backbone, single pass, and lightweight architecture with an end-to-end optimization manner [59]; hence, in [60], they used it for traffic sign recognition. In addition, it has the capability to predict objects with 45 frames per second (FPS) and can achieve 155 FPS [61]. The main limitation of YOLO is that it cannot detect small and numerous objects in specified regions [62]. Furthermore, it is difficult for YOLO to work on multiple scales [63]. The major change from other networks is that, despite using classifiers to perform detection, they practice object detection as a regression assignment to spatially separate bounding boxes and related class probabilities. In [64], the detection demonstration is improved in YOLO by using the pass-through layer that combines feature maps from the front layers to get fine-grained features to reduce the negative effect of model size.

The probabilities are predicted right away from images during solitary evaluation through only using neural networks [65]. YOLOv1 was followed by YOLOv2, published after performing several developments, such as Dark-net 19 convolutional layers architecture, advanced resolution classifier, bounded boxes, and various usages of the dataset for preparation, such as PASCAL VOC and COCO classifier,r making it

quicker and more accurate. In [66], Pothole detection using YOLOv2 object detection is done. YOLOv2 is an improved version of YOLO and outperforms significantly in real-time with maintained inference speed. YOLOv2 followed a powerful and strong deep CNN backbone architecture, which could be trained on high-resolution images such as 448x448. YOLOv2 showed a better anchor strategy for training data using k-means clustering. YOLOv2 used batch-normalization and multilevel training methods, attained state-of-the-art results in object detection, and reduced difficulties in optimized localization [67]. In [68], an improved YOLOv2 has been proposed for object detection to solve the limitations of the basic version on small object detection and large parameters in the model. In this way, standard convolution is replaced by depth-wise separable convolution in YOLOV2. Additionally, FPN is used as a detection model to enhance small object detection on multi-level images compared to the previous feature fusion method, and it also reduced the number of parameters by 78.83% [69].

Another updated YOLO version named YOLOv3 with little improvement has been proposed in [70]. It has increased the number of steps without speed loss. YOLOv3 computes as fast as 22ms on 320×320 and results in 28.2 mAP. YOLOv3 performed better than SSD but faster and attained 57.9mAP@50 in 51 ms [71]. Real-time vision and object detection and its process on an embedded system with limited memory and low computation capabilities is a challenging task [72]. To compensate for this issue, the authors proposed a lightweight Darknet-53-based network to reduce the parametric size object detection model by 16%. In [73], YOLO is developed to optimize the accuracy and speed of detection of malaria pathogens with minimal resources. Moreover, the small-scale detection model without loss of accuracy uses a multi-scale feature-pyramid to give a better performance called mini-YOLOv3 [74]. As compared to basic YOLOv3, mini-YOLOv3 is only 23% of the previous model with fewer parameters. YOLOv3 was trained on the COCO dataset comprising 80 labels [75]. There are clear differences among various versions of YOLO [76]-[83]. YOLOv3 uses logistic classifiers for respective classes by assigning scores to the objects [84].

YOLOv4 was introduced in 2020 [85], where a bag of freebies (BoF) and a bag of specials (BoS) were used. The BoF makes progress in accuracy without affecting time [86]. By increasing cost, the BoS increases the accuracy of object detection [87]. In [88] YOLOv5 algorithm was introduced, and where PyTorch library of the Python language was used during the development of this algorithm. The model is improved by implementing a data augmentation technique, and it has the feature of auto-cultured bounding case anchor [89].

The initial Single Shot Detector (SSD) was given in [90], and afterward used VGG16 as a rudimentary network because it has better image classification. In [91], an efficient face mask detector was built using a Single Shot Detector (SSD). Stationary candidate objects were classified using SSD [92] to detect suspected persons near some objects of point of interest, such as backpacks and handbags, at a certain time, for surveillance purposes. Stationary objects were removed from the detection process to avoid an alarm since they were not required to be detected. The fundamental property of SSD is the smearing of minor convolutional filters to extract maps. These maps are used to calculate class scores and offsets for a static set of default bounding boxes [93]. In [94], a novel single-shot refinement neural network was publicized for firm and precise 3D object detection from the raw LiDAR point cloud. It is observed that the SSD model has been enhanced by numerous scholars [95]. VGG16 can be used as an item classifier for SSD, MobileNet, or ResNet [96]. The major difference between YOLO and SSD is that YOLO takes a fully connected layer while the SSD swaps it with a convolutional filter [97].

R-CNN is a two-stage detector algorithm [98], where the workflow in R-CNN is divided into three main steps, i.e., proposal generation, feature detection, and image class classification. R-CNN creates a random proposal set using a Selective Search for each image, and its structure is used to reject regions that can be identified on the basis of previous regions. Additionally, each input proposal is resized into a fixed value of region size and then encoded for feature extraction using a Support Vector Machine (SVM) [99]. Proposal generation and its methods are important to be considered for the algorithm's performance [100]-[103]. In [104], R-CNN is used for synthetic aperture radar (SAR) target detection. In [105], Faster R-CNN is used for sea surface object detection. In [106], Faster R-CNN is used for polyp detection for colon cancer. In [107], a detection method of pulmonary embolism based on Faster R-CNN is proposed. In [108], Pulmonary Nodule Detection Based on Faster R-CNN with Adaptive Anchor Box is obtained. In [109], Cascade Faster R-CNN Detection for Vulnerable Plaques in OCT Images is done. In [110], Rapid Detection of Rice Disease Based on Faster R-CNN is done. In [111], Faster R-CNN is developed for the same object retrieval. In [112], Faster R-CNN

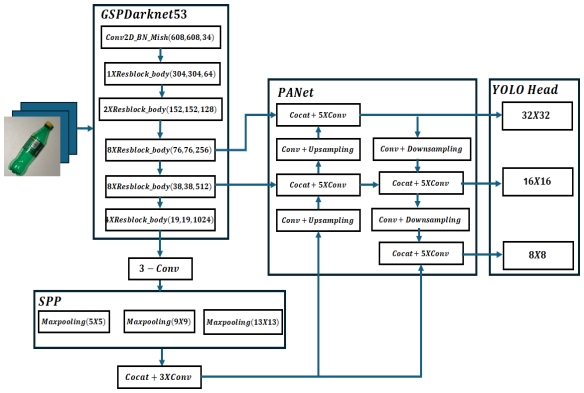

Fig. 1: Architectural overview of YOLOv4 algorithm [125]

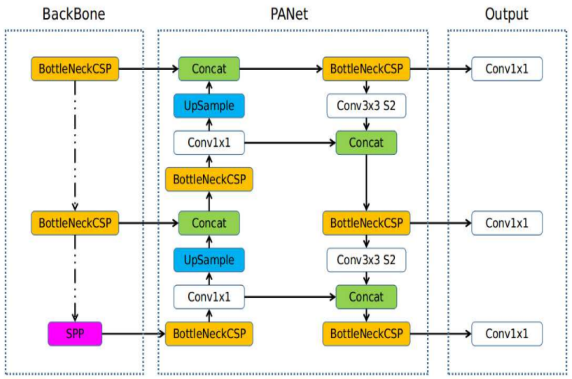

Fig. 2: Architectural overview of YOLOv5 algorithm [133]

for failed satellite components detection is done. In [113], adaptive defect detection for 3-D printed lattice structures based on Faster R-CNN is done. In [114], fast and accurate craniomaxillofacial landmark detection via 3D Faster R-CNN is achieved.

3 Material and Methods

3.1 Algorithms and Architecture

YOLOv4: It is an object detection algorithm with the YOLO framework [115], that aims to improve object detection accuracy and speed by merging various computer vision research [116]. YOLOv4-tiny, YOLOv4-small, YOLOv4-medium, YOLOv4-large, and YOLOv4-xlarge are its various models, each offering a trade-off between accuracy and speed. Some key features and improvements introduced in YOLOv4 are as follows [117-125]:

- YOLOv4 has a modified CSPDarknet53 backbone architecture, a Darknet design used in previous YOLO versions. This backbone uses convolutional and shortcut layers for feature extraction.
- Feature pyramid network (FPN) in YOLOv4 enhances detection at various scales for robust multi-scale object detection.
- Path Aggregation Network (PANet) is used to improve spatial awareness of the model by helping it to get related information.
- The problem of gradient saturation is improved using a Mish activation function, which enhances convergence speed and accuracy.
- It uses the GIoU loss function, and a cosine annealing scheduler for learning rate adjustment and data augmentation methods such as mosaic and random shapes.
- It adopts advanced training techniques such as Bag of Freebies (BoF) and Bag of Specials (BoS), to optimize label smoothing, and grid sensitivity for better performance

YOLOv4 is a complex algorithm having different variations with specific features. It also requires high-level computational resources for training and implication. The overview of the YOLOv4 algorithm is shown in Fig. 1.

YOLOv5: It is one of the most recent object detection algorithms used for real-time object detection. It is an enhanced version of YOLOv4 [126]. In order to get better detection accuracy by using a more efficient and flexible PyTorch deep learning framework [127], designed such that it is user-friendly [128]. It has further sub-models such as YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, with suffixes indicating small, medium, large, and extra-large [129]. Each model has its own accuracy and speed level, and thus can be chosen as per requirement. YOLOv5 has many advanced features that include the use of anchor-free bounding box prediction, better data augmentation techniques, and training strategies. It also has various backbones, such as CSPDarknet, EfficientNet, and ResNet, providing flexibility in choosing the underlying architecture [130]. The YOLOv5 framework provides pre-trained models, and it is easier to start an object detection task [131]. However, YOLOv5 also has limitations such as dataset quality, model fine-tuning, and deployment scenarios to achieve optimal results for specific use cases [132]. The architectural overview of the YOLOv5 algorithm is shown in Fig. 2.

Single shot detection (SSD): It is an object detection algorithm that was introduced by Wei Liu et al., in 2016 [134]. SSD is designed to perform real-time object detection in images by predicting class labels and bounding box coordinates for multiple objects in a single pass through a convolutional neural network (CNN) [135]. Some key features and characteristics of

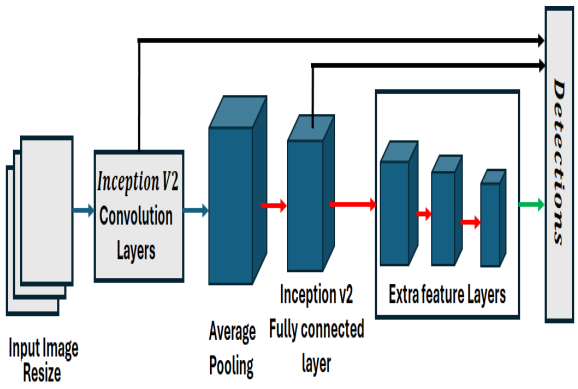

Fig. 3: Architectural overview of SSD algorithm [144]

SSD are as follows [136-144]:

- SSD uses a series of convolutional layers of different sizes to capture features at multiple scales, having different spatial resolutions, to allow the model to detect objects of various sizes.
- Pre-defined bounding boxes of different aspect ratios and sizes are used to predict object locations and sizes. Predictions at different scales are performed. For each scale, the model forecasts class probabilities for each anchor box and adjusts the coordinates of the anchor boxes to match the ground truth objects in the image. Furthermore, it can detect multiple object classes simultaneously.
- Instead of using a fixed set of anchor boxes, SSD generates default boxes at different scales and aspect ratios dynamically based on the aspect ratio and size of the ground truth objects present in the training dataset. This allows the model to adapt to different object shapes and sizes.
- The training of SSD involves optimizing a combined loss function that incorporates both classification and localization losses, to measure the accuracy of class predictions and the accuracy of predicted bounding box coordinates, respectively.

SSD has been widely adopted in various computer vision applications due to its capacity to balance speed and accuracy in real-time object detection tasks. It has become a popular choice for scenarios that need fast and efficient object detection, such as video analysis, autonomous driving, and surveillance systems [143]. The overview of the SSD algorithm is shown in Fig. 3.

Faster-RCNN: A faster Region-based Convolutional Neural Network is an algorithm based on an extension of the original R-CNN framework in terms of addressing some of its limitations, particularly in terms of accuracy and speed [145]. Key features and components of Faster R-CNN are as follows [146-154]:

- It introduces a Region Proposal Network (RPN) for the convolutional feature maps of the input image, which generates a set of region proposals, which are potential bounding box locations, containing objects of interest.
- Faster R-CNN shares the convolutional layers between the RPN and the subsequent object detection network to reduce computation and make the network trained end-to-end.
- RoI (Region of Interest) pooling is generated through Faster R-CNN applies RoI pooling, which extracts fixed-size feature vectors from the convolutional feature maps for each proposal, then feeds them into fully connected layers for subsequent classification and bounding box regression.
- Classification and bounding box regression are simultaneous, using the RoI features to predict class probabilities and refine the coordinates of the bounding boxes for each proposed region.
- Training with backpropagation is performed to generate accurate region proposals, then the shared convolutional layers and the subsequent layers are fine-tuned for object classification and bounding box regression using the region proposals.
- The training of Faster R-CNN involves optimizing a combined loss function that includes a classification loss (usually cross-entropy loss) and a bounding box regression loss (typically based on the smooth L1 loss). The losses are computed for both the RPN and the object detection network.

Faster R-CNN achieves higher accuracy compared to its predecessor, R-CNN, and its variants, such as Fast R-CNN, by eliminating the need for external region proposal methods and integrating the proposal generation process within the network. It combines the benefits of region proposal techniques with the efficiency of shared convolutional layers, making it widely adopted for various object detection tasks, including instance segmentation and object localization in both images and videos. The overview of the R-CNN algorithm is shown in Fig. 4.

3.2 Object Detection

In this work algorithms given in Section 3.1 are used for object detection. These algorithms are used on the same images and videos in the same manner in order to compare them. An image or video sample is fed to the detection model as input, which outputs the images with labels and bounded boxes. The provided sample is first pre-processed for object detection to make it suitable for the object detection model. After preprocessing, a pre-trained model is loaded. Loading a pre-trained model for object detection typically involves

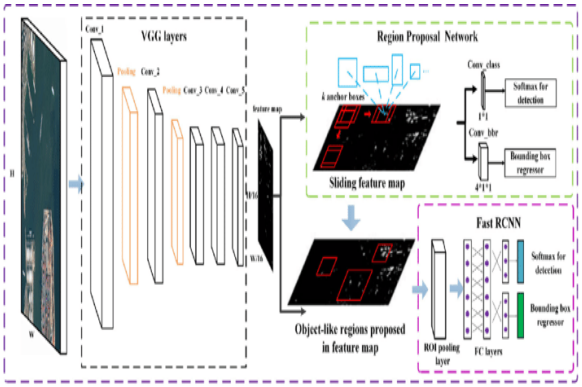

Fig. 4: Architectural overview of R-CNN [154]

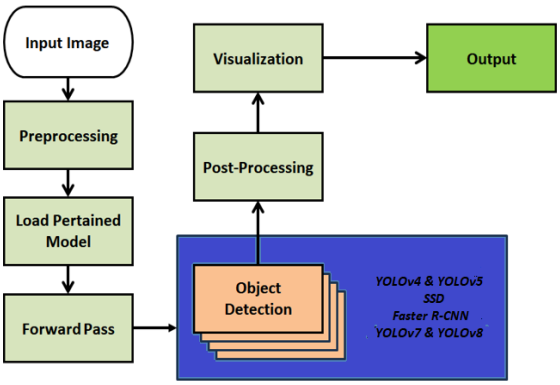

Fig. 5: Flowchart for object detection using various algorithms

using a deep learning framework like TensorFlow or PyTorch.

After loading the pre-trained model, a forward pass is applied. The pre-trained model performs detection on images or videos by passing the input data through the model and interpreting the output. The exact process for performing inference will depend on the specific framework and model used. In object detection, the forward pass refers to the process of passing an input image through a pre-trained neural network to obtain predictions for the presence and location of objects in the image. The forward pass involves feeding the input image through the model's layers, computing the necessary transformations and operations, and finally obtaining the output predictions.

Afterward, object detection is performed to identify and locate objects of interest within an image or a video using deep learning models of Section 3.1. The detection algorithm draws bounding boxes around predicted objects to indicate their precise locations. Post-processing is also required to refine the results by using various techniques to filter, refine, and organize the detected objects. It is also needed to remove any chance of duplication or unwanted results.

After post-processing, the visualization of object detection results and the performance of an algorithm are tested. Most visualization techniques are implemented using computer vision libraries such as OpenCV. The output of object detection is a set of predictions that indicate the presence, location, and class of objects within an input image or video frame. The output format may vary slightly depending on the specific object detection model and the deep learning framework used. The output is usually in the form of a list of predictions, with each prediction containing the bounding box coordinates, class label, and confidence score. Depending on the implementation and requirements, additional information may also be included in the output, such as key points for pose estimation, mask segmentation for instance segmentation, or tracking IDs for multi-object tracking. The flowchart for object detection is shown in Fig. 5.

3.3 Evaluation Approach

Primarily, in this paper, each algorithm will be assessed for real-time videos. For evaluation of object detection, the accuracy of the algorithm comparison through mean average precision and frames per second will be done. In terms of exactness, there are numerous diverse approaches utilized to assess the exactness of detection; however, mAP and FPS are the most important properties. Intersection over union (IoU) is used to evaluate the performance of object detection, given as;

$$IoU = \frac{Areaofoverlap(A_{gt} \cap A_p)}{AreaofUnion(A_{gt} \cup A_p)} \tag{1}$$

where A_{gt} is the ground-truth bounding box, A_p is the predicted bounding box, and the threshold of IoU is 0.5 to classify whether either prediction is a true positive or a false positive.

With TP, as be true positive value, TN, as the true negative value, FP, as the false positive value, and FN be the false negative value, the precision value P is given as:

$$P = \frac{TP}{TP + FP} \tag{2}$$

The recall value R, which is the ratio of the number of true positives divided by the sum of true positives and false negatives, is given as:

$$P = \frac{TP}{TP + FN} \tag{3}$$

Average precision AP, whose value falls in the range of 0-1, the average accuracy rate, which is the integral of the precision index to the recall index, is given as:

$$AP = \int_1^0 (P.R)dR \tag{4}$$

The value of mAP is the average accuracy of the mean and is given as:

$$mAP = \frac{1}{Q_R} \sum_{q=Q_R} AP(q) \tag{5}$$

where Q_R is the number of categories. On the other hand, frame per second (FPS) is a typical assessment measure utilized to determine how quickly it is for a proposed organized demonstration to distinguish object outlines per second on average. In addition to mAP and FPS, efficiency and latency are two other evaluation metrics for object detection. Efficiency indicates the achievement of high accuracy with low computational resources.

4 Result and Discussion

The comparison of object detection is done using YOLOv4, YOLOv5, SSD, and Faster-RCNN in a Python environment. Object detection using various images and real-time videos is performed. The comparison of the object detection for various algorithms will be achieved using each image or video. In the end, the performance of each algorithm in terms of speed and accuracy is verified along with YOLOv7 and YOLOv8.

4.1 Single Image Object Detection

In Fig. 6, object detection results for various algorithms under consideration are shown. Each algorithm, i.e., YOLOv5, YOLOv4, SSD, and Faster RCNN, was tested using a single picture. Five different pictures are used with various locations, with people, cars, and mobile phones taken as objects to be detected. It can be observed in Fig. 6 that all the objects are detected the objects accurately except Faster-RCNN, which means that they are good in terms of accuracy. In Table 1 properties such as accuracy, execution time, and speed are taken into consideration for single image input via various algorithms/ Whereas Fig.8 represents the bar chart, which compares four object detection models (YOLOv5, YOLOv4, SSD, and Faster-RCNN) in terms of two properties, accuracy and response time. In terms of accuracy, YOLOv4 has the highest accuracy, followed by YOLOv5, SSD has moderate accuracy, and Faster-RCNN has the lowest accuracy among the four models. In terms of response time, Faster-RCNN takes the most time to perform detection, followed by YOLOv4, SSD performs relatively fast, and YOLOv5 is the fastest in terms of detection time.

This comparison suggests that while YOLOv4 offers the best accuracy, it trades off by taking more time, whereas YOLOv5 is a good balance between speed and accuracy. Faster RCNN is slow but might be preferred in scenarios where accuracy isn't the primary concern. The accuracy metric is crucial for performance comparison because it is determined by the number of quality training samples, threshold values, and the model's parameters, which ultimately reflect an algorithm's precision and overall capabilities. In our analysis, YOLOv4 demonstrated higher accuracy compared to YOLOv5, Faster R-CNN, and SSD. Faster R-CNN showed a tendency to miss some instances, which can be attributed to a high number of false negatives (FN), recorded at 150, thus reducing its detection accuracy. This highlights the importance of accuracy in evaluating an algorithm's robustness. Regarding processing time, we measured the execution time for detecting objects in a single frame. YOLOv5 emerged as the fastest algorithm, with detection times ranging from 0.12 to 0.2 seconds per frame. In contrast, Faster R-CNN required around 1 second per frame, which highlights the substantial performance gap in processing speed between these models. Notably, the execution time was consistent across various image sizes, showing no significant dependency on the image dimensions.

4.2 Video outputs

In Fig. 7, snapshots for object detection in a video for various algorithms under consideration are given. Faster-RCNN is not taken into consideration for real-time object detection because of its poor performance in terms of accuracy. YOLOv5, YOLOv4, and SSD are tested using the same video with people, cars, and objects. It can be observed in Fig. 7 from the video output results that YOLOv4 has the best accuracy compared to YOLOv5 and SSD.

In Table 2, the accuracy, execution time, and speed are taken into consideration for the video. Fig.9 represents a bar chart comparing mAP (mean Average Precision) and Time across different object detection models (YOLOv5, YOLOv4, and SSD). It can be observed that YOLOv4 (green) has the highest mAP, indicating it is the most accurate among the three models for object detection. YOLOv5 (blue) has slightly lower mAP compared to YOLOv4, but still

TABLE 1: One image output, visual results based on accuracy

Property	yolov5	yolov4	SSD	Faster-RCNN
Accuracy	Less than yolov4 but more than SSD and Faster-RCNN	Highest	Less than yolov4 and yolov5 but more than Faster-RCNN	Lowest
Time	0.12 ˜ 0.20 (s/frame)	0.86 ˜ 0.94 (s/frame)	0.17 ˜ 0.23 (s/frame)	1s/frame
Speed	Fastest	Slower than yolov5 and SSD, but faster than Faster-RCNN	Faster than yolov4 and Faster-RCNN but slower than yolov5	Slowest

TABLE 2: Comparison based on video detection

Property	yolov5	yolov4	SSD
Accuracy	Less than yolov4 but more than SSD	More than yolov5 and SSD	Less than yolov4 and yolov5
Time	0.14 ˜ 0.22 (s/frame)	0.87 ˜ 0.96 (s/frame)	0.18 ˜ 0.25 (s/frame)
Speed	Faster than yolov4 and SSD	Slower than yolov5 and SSD	Faster than yolov4 but slower than yolov5

Fig. 6: Object detection results on images using various algorithms; (a) YOLOv5, (b) YOLOv4, (c) SSD, and (d) Faster-RCNN

performs well in terms of precision. While SSD (red) has the lowest mAP, suggesting it is less accurate in detecting objects compared to YOLOv5 and YOLOv4. For execution time for detection, the YOLOv4 (green) takes the most time for object detection, which implies it is slower but highly accurate. YOLOv5 (blue) is much faster than YOLOv4, and its time is the lowest among the three models, making it the fastest model in terms of processing time. While SSD (red) also has a shorter detection time, faster than YOLOv4, but slower than YOLOv5. This means that the YOLOv4

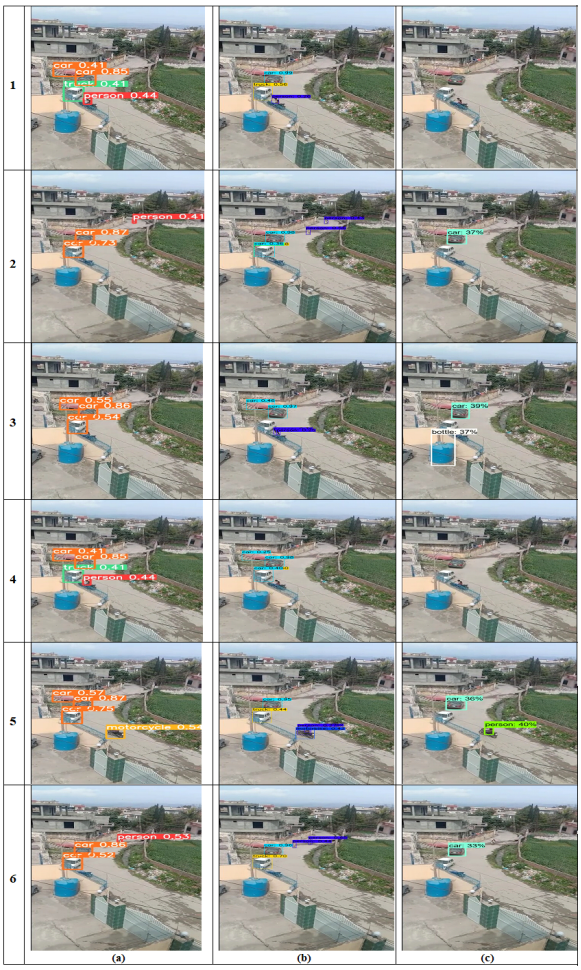

Fig. 7: Object detection results on video frames using various algorithms; (a) YOLOv5, (b) YOLOv4, and (c) SSD

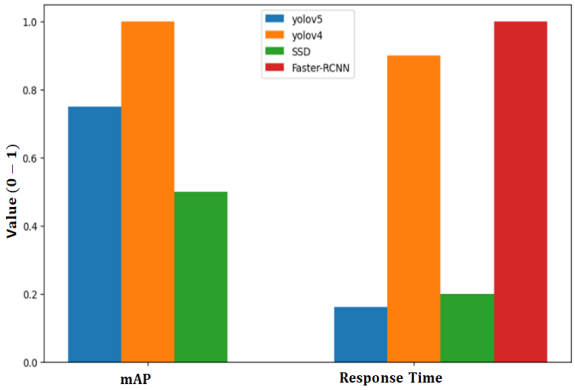

Fig. 8: Performance comparison of object detection models

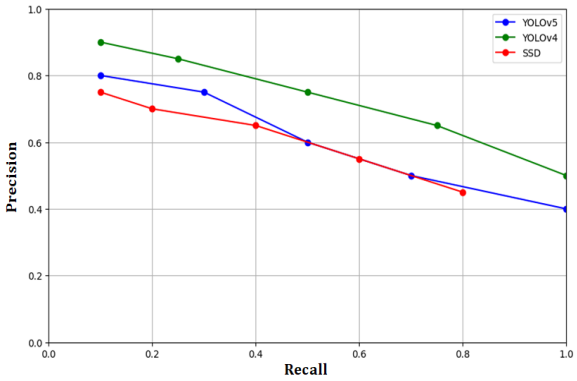

Fig. 10: Precision-Recall curves for three object detection models

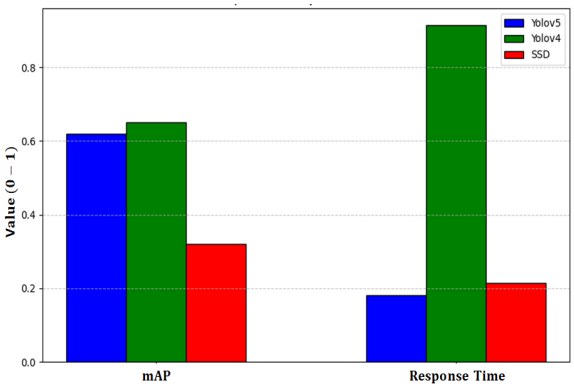

Fig. 9: mAP and time comparison of YOLOv5, YOLOv4, and SSD models

provides the best accuracy (highest mAP) but at the cost of longer detection times. YOLOv5 balances both accuracy and speed, making it an efficient choice for real-time applications. However, SSD is a fast model but sacrifices some accuracy for speed.

This comparison suggests that YOLOv4 is ideal when accuracy is the priority, while YOLOv5 is preferable when fast processing times are more critical, especially in real-time scenarios. SSD might be chosen when speed is important, but with an acceptable compromise in precision. The accuracy can be compromised because it has shown satisfactory performance in accuracy. The mAP of all the algorithms is also given, where it is observed that YOLOv4 is more accurate compared to its boost version YOLOv5, and also greater specific than SSD. Furthermore, frames per second (FPS) is a popular assessment criterion used to measure how speedy it is for a proposed community model to hit upon items frames per second on common. Additionally, it has been called body rate or frame frequency. If the rate is better, it means that it has a better overall performance to deal with more pixels every second. The FPS of those three algorithms is given in Table 3. Yolov5 can system 58 frames consistent with second, Yolov4 can do 54 frames in step with second, while SSD is capable of processing 56 frames according to second. The FPS is probably numerous depending on the overall performance of hardware gadgets like the GPU, CPU, and so on. However, it can be declared that the Yolov5 model detection has a better speed than Yolov4 and SSD. Those algorithms can also compare the bottom of the time taken to process one frame. Outcomes show that YOLOv5 is the quickest among the two different algorithms in this domain.

Fig.10 shows the Precision-Recall curves for three object detection models: YOLOv5, YOLOv4, and SSD. The Recall is given on the x-axis for the proportion of actual positive instances correctly identified by the model. Precision is given on the y-axis for the proportion of predicted positive instances that were actually positive. The YOLOv5 (Blue) curve maintains a relatively high precision across a broad range of recall values, indicating that it has a good balance between precision and recall. The YOLOv4 (Green) curve shows high precision initially but tends to decrease as recall increases, indicating that it may struggle to maintain precision at higher levels of recall. While SSD (Red) has the lowest precision across most recall levels, suggesting it may be less effective than the other two models in accurately detecting objects.

4.3 Comparison with Recent Algorithms

The comparison of YOLOv4, YOLOv5, and SSD (Single Shot MultiBox Detector) is further performed with YOLOv7 and YOLOv8 [155-157], based on performance and usability, evaluated on the same environment and machine. This comparison includes

TABLE 3: Comparison based on mAP, FPS, Time

Measure	Yolov5	Yolov4	SSD
mAP	0.62	0.65	0.32
FPS	58	54	56
Time	0.14 ˜ 0.22 (second / per frame)	0.87 ˜ 0.96 (second / per frame)	0.18 ˜ 0.25 (second / per frame)

real-time video stream processing and feature calculation. Evaluation in terms of architecture and design of these object detection models suggests that YOLOv4 and YOLOv5 have a CNN-based design with CSP-enhanced backbones, PANet-style, and standard YOLO detection features. YOLOv7, on the other hand, is an enhanced backbone and a task-aligned prediction for improved accuracy. YOLOv8 uses anchor-free detection; therefore, it is more flexible than other YOLO models. SSD has a simpler architecture with an FPN neck and MultiBox detection head; therefore, it is fast but less accurate compared to YOLO-based algorithms. In Table 4, a comparison of YOLOv4, YOLOv5, YOLOv7, YOLOv8, and SSD, based on their performance, accuracy, training, and key features, is given. In terms of performance, mean-average precision (mAP), speed (FPS), efficiency, and latency are considered, and it is concluded that YOLOv8 is the best choice for real-time applications, offering the highest accuracy, fastest speed, lowest latency, and best optimization. While YOLOv7 is a close second, it is highly efficient and fast. YOLOv5 is still a great balance of speed and accuracy for many applications.

In terms of training and deployment, it is concluded in Table 4 that YOLOv8 is the easiest to train, offers the most pre-trained model variants, and supports multiple frameworks (PyTorch, ONNX, TensorRT, OpenVINO, CoreML), making it the best choice for deployment. YOLOv5 is also easy to train and provides strong support for custom datasets. However, it has fewer deployment options than YOLOv8. YOLOv7 and YOLOv4 require moderate effort for training, with YOLOv7 being more optimized for performance. In terms of key features, the YOLOv8 is observed to be the most versatile model, supporting instance segmentation, pose estimation, classification, and both anchor-based and anchor-free detection, making it ideal for multi-task applications. YOLOv7 is the only other model that supports pose estimation, but it lacks instance segmentation and anchor-free detection. YOLOv5 and YOLOv4 are limited to object detection only, with no support for segmentation or pose estimation. SSD is the least feature-rich, only supporting object detection and lacking advanced capabilities. Overall, YOLOv8 is the best choice for applications requiring multiple tasks like detection, segmentation, and pose estimation. Therefore, it may be concluded that YOLOv8 is the most versatile model, excelling in real-time detection, edge device compatibility, high-accuracy applications, and multi-task flexibility (detection, segmentation, classification). YOLOv7 is also a strong choice for real-time detection and high-accuracy tasks, but is less optimized for edge devices and lacks segmentation/-classification support. YOLOv5 is well-optimized for real-time detection and edge devices, making it a balanced choice for lightweight applications.

5 Conclusion

In this paper, a survey and comparison of various deep learning algorithms for object detection have been provided, focusing on YOLOv5, YOLOv4, SSD, and Faster RCNN for both single images and video frames. It is observed that both versions of YOLO outperform SSD and Faster RCNN in accuracy, as some objects remain undetected by the pre-trained models of the latter algorithms. YOLOv4 and YOLOv5 show better accuracy, while SSD demonstrates decent speed compared to YOLOv4. YOLOv5 emerges as the fastest algorithm overall. If accuracy is the priority, YOLOv4 is recommended, but for real-time applications where speed is critical, YOLOv5 is a better option. The trade-off between speed and accuracy depends on the user's requirements. On the basis of accuracy, YOLOv4 stands out compared to YOLOv5, Faster-RCNN, and SSD. Faster-RCNN is not suitable for real-time detection due to its poor performance in both speed and accuracy. However, future improvements in these models could enhance their capabilities by leveraging deep learning techniques and refining the algorithms' layers. Then the paper further compared various deep learning algorithms for object detection, focusing on YOLOv4, YOLOv5, YOLOv7, YOLOv8, and SSD for both single images and video frames. The results indicate that YOLO models significantly outperform SSD in both accuracy and speed, as some objects remain undetected by SSD's pre-trained models. Among the YOLO versions, YOLOv8 achieves

TABLE 4: Comparison of various object detection models

Feature	YOLOv4	YOLOv5	YOLOv7	YOLOv8	SSD
Comparison based on Speed and Accuracy					
mAP@50 (COCO)	˜50-55%	˜55-56%	˜56-57%	˜57-58%	˜42-45%
Speed (FPS)	Slower	Faster	Faster than YOLOv5	Fastest	Slower
Efficiency	High but heavier	Efficient	High efficiency	Most optimized	Moderate
Latency	High	Low	Moderate	Lowest	Higher than YOLO
Comparison based on Training and Deployment					
Ease of Training	Moderate	Easy	Moderate	Easiest	Difficult
Pretrained Models	COCO	COCO, custom	COCO, custom	More pre-trained variants	COCO, VOC
Framework Support	Darknet, TensorFlow, OpenCV	PyTorch	PyTorch	PyTorch, ONNX, TensorRT, OpenVINO	TensorFlow, PyTorch
Exportability	ONNX, TensorRT	ONNX, TensorRT	ONNX, TensorRT	ONNX, TensorRT, OpenVINO, CoreML	ONNX, TensorFlow, TensorRT
Comparison based on Key Features					
Instance Segmentation	No	No	No	Yes	No
Pose Estimation	No	No	Yes	Yes	No
Multi-tasking	Object detection only	Object detection	Object detection, pose estimation	Object detection, segmentation, classification	Object detection
Anchor-Free Support	No	No	No	Yes	No

the highest accuracy, followed by YOLOv7, YOLOv5, and YOLOv4. However, YOLO models have several limitations, which include their struggle with detecting small or obscured objects with complex backgrounds. Additionally, the computational cost associated with high-performing models like YOLOv8 is another problem that can be faced during the detection process.

References

[1] Y.-J. Liang, X.-P. Cui, X.-H. Xu, and F. Jiang, "A review on deep learning techniques applied to object detection," in Proc. 7th Int. Conf. Inf. Sci. Control Eng. (ICISCE), Changsha, China, 2020, pp. 120–124, doi: 10.1109/ICISCE50968.2020.00035.

[2] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High quality object detection and instance segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 5, pp. 1483–1498, May 2021, doi: 10.1109/TPAMI.2019.2956516.

[3] O. Chapelle, P. Haffner, and V. N. Vapnik, "Support vector machines for histogram-based image classification," IEEE Trans. Neural Netw., vol. 10, no. 5, pp. 1055–1064, Sep. 1999, doi: 10.1109/72.788646.

[4] E. U. Samani, X. Yang, and A. G. Banerjee, "Visual object recognition in indoor environments using topologically persistent features," IEEE Robot. Autom. Lett., vol. 6, no. 4, pp. 7509–7516, Oct. 2021, doi: 10.1109/LRA.2021.3099460.

[5] W. Cao, J. Yuan, Z. He, Z. Zhang, and Z. He, "Fast deep neural networks with knowledge guided training and predicted regions of interests for real-time video object detection," IEEE Access, vol. 6, pp. 8990–8999, 2018, doi: 10.1109/ACCESS.2018.2795798.

[6] M. Wu, C. Zhang, J. Liu, L. Zhou, and X. Li, "Towards accurate high-resolution satellite image semantic segmentation," IEEE Access, vol. 7, pp. 55609–55619, 2019, doi: 10.1109/ACCESS.2019.2913442.

[7] J. Zhao, N. Zhang, J. Jia, and H. Wang, "Digital watermarking algorithm based on scale-invariant feature regions in non-subsampled contourlet transform domain," J. Syst. Eng. Electron., vol. 26, no. 6, pp. 1309–1314, Dec. 2015, doi: 10.1109/JSEE.2015.00143.

[8] X. Wang, "Moving window-based double Haar wavelet transform for image processing," IEEE Trans. Image Process., vol. 15, no. 9, pp. 2771–2779, Sep. 2006, doi: 10.1109/TIP.2006.877316.

[9] T. Zhang et al., "HOG-ShipCLSNet: A novel deep learning network with HOG feature fusion for SAR ship classification," IEEE Trans. Geosci. Remote Sens., vol. 60, pp. 1–22, 2022, Art. no. 5210322, doi: 10.1109/TGRS.2021.3082759.

[10] X. Wang, X. Lv, L. Li, G. Cui, and Z. Zhang, "A new method of speeded up robust features image registration based on image preprocessing," in Proc. Int. Conf. Inf. Syst. Comput. Aided Educ. (ICISCAE), Changchun, China, 2018, pp. 317–321, doi: 10.1109/ICISCAE.2018.8666894.

[11] W. Ding, Y. Tong, Q. Zhang, and D. Yang, "Image and video quality assessment using neural network and SVM,"

Tsinghua Sci. Technol., vol. 13, no. 1, pp. 112–116, Feb. 2008, doi: 10.1016/S1007-0214(08)70018-X.

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, Jan. 2014.

[14] A. Saudergiene, B. Porr, and F. Worgotter, "Biologically inspired artificial neural network algorithm which implements local learning rules," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Vancouver, BC, Canada, 2004, pp. V–V, doi: 10.1109/ISCAS.2004.1329586.

[15] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in Proc. Int. Conf. Comput. Vis. (ICCV), Nov. 2011, pp. 991–998.

[16] P. Kumar, S. Batchu, N. Swamy, and S. R. Kota, "Real-time concrete damage detection using deep learning for high rise structures," IEEE Access, vol. 9, pp. 112312–112331, 2021, doi: 10.1109/ACCESS.2021.3102647.

[17] E. Güney, C. Bayilmiş, and B. Çakan, "An implementation of real-time traffic signs and road objects detection based on mobile GPU platforms," IEEE Access, vol. 10, pp. 86191–86203, 2022, doi: 10.1109/ACCESS.2022.3198954.

[18] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in Proc. Neural Inf. Process. Syst. (NIPS), 2016, pp. 379–387.

[19] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 10012–10022.

[20] K.-J. Kim et al., "Performance enhancement of YOLOv3 by adding prediction layers with spatial pyramid pooling for vehicle detection," in Proc. IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), Nov. 2018, pp. 1–6.

[21] H. Wu et al., "Sequence level semantics aggregation for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 9217–9225.

[22] Y. Chen, Y. Cao, H. Hu, and L. Wang, "Memory enhanced global-local aggregation for video object detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 10337–10346.

[23] S. Wang, Y. Zhou, J. Yan, and Z. Deng, "Fully motion-aware network for video object detection," in Proc. Eur. Conf. Comput. Vis., Cham, Switzerland: Springer, 2018, pp. 542–557.

[24] X. Chen, J. Yu, and Z. Wu, "Temporally identity-aware SSD with attentional LSTM," IEEE Trans. Cybern., vol. 50, no. 6, pp. 2674–2686, Jun. 2020.

[25] M. Zhu and M. Liu, "Mobile video object detection with temporally-aware feature maps," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 5686–5695.

[26] C. Zhang and J. Kim, "Video object detection with two-path convolutional LSTM pyramid," IEEE Access, vol. 8, pp. 151681–151691, 2020.

[27] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, "Relation distillation networks for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 7023–7032.

[28] H. Deng et al., "Object guided external memory network for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 6678–6687.

[29] M. Fujitake and A. Sugimoto, "Temporal feature enhancement network with external memory for object detection in surveillance video," in Proc. 25th Int. Conf. Pattern Recognit. (ICPR), Jan. 2021, pp. 7684–7691.

[30] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, Dec. 2015.

[31] L. Wen et al., "UA-DETRAC: A new benchmark and protocol for multiobject detection and tracking," Comput. Vis. Image Understand., vol. 193, Apr. 2020, Art. no. 102907.

[32] T.-Y. Lin et al., "Feature pyramid networks for object detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 936–944.

[33] N. Carion et al., "End-to-end object detection with transformers," in Proc. Eur. Conf. Comput. Vis., Cham, Switzerland: Springer, 2020, pp. 213–229.

[34] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 3057–3065.

[35] K. Kang et al., "Object detection in videos with tubelet proposal networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 889–897.

[36] K. Kang et al., "T-CNN: Tubelets with convolutional neural networks for object detection from videos," IEEE Trans. Circuits Syst. Video Technol., vol. 28, no. 10, pp. 2896–2907, Oct. 2018.

[37] W. Han et al., "Seq-NMS for video object detection," in Proc. Workshops Int. Conf. Learn. Represent., 2016, pp. 1–4.

[38] X. Zhu et al., "Towards high performance video object detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7210–7218.

[39] J. Kim et al., "Video object detection using object's motion context and spatio-temporal feature aggregation," in Proc. 25th Int. Conf. Pattern Recognit. (ICPR), Jan. 2021, pp. 1604–1610.

[40] C. Guo et al., "Progressive sparse local attention for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 3909–3918.

[41] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in Proc. Int. Conf. Learn. Represent., 2018, pp. 1–10.

[42] G. Bertasius, L. Torresani, and J. Shi, "Object detection in video with spatiotemporal sampling networks," in Proc. Eur. Conf. Comput. Vis., Cham, Switzerland: Springer, 2018, pp. 331–346.

[43] P. Goyal et al., "Accurate, large minibatch SGD: Training ImageNet in 1 hour," arXiv preprint, arXiv:1706.02677, 2017.
[

[44] K. He, R. Girshick, and P. Dollár, "Rethinking ImageNet pre-training," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 4918–4927.

[45] M. Shvets, W. Liu, and A. Berg, "Leveraging long-range temporal relationships between proposals for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 9756–9764.

[46] L. He et al., "End-to-end video object detection with spatial-temporal transformers," in Proc. 29th ACM Int. Conf. Multimedia, Oct. 2021, pp. 1507–1516.

[47] T. Gong et al., "Temporal ROI align for video object recognition," in Proc. AAAI Conf. Artif. Intell., vol. 35, no. 2, pp. 1442–1450, 2021.

[48] Y. Cui et al., "TF-blender: Temporal feature blender for video object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 8138–8147.

[49] R. Kundu, S. Das, and S. K. Mishra, "A comprehensive review of single and double stage object detection algo-

rithms in self driving cars," in Proc. IEEE 7th Int. Conf. Convergence Technol. (I2CT), Mumbai, India, 2022, pp. 1–8, doi: 10.1109/I2CT54291.2022.9824384.

[50] S. V. Mahadevkar et al., "A review on machine learning styles in computer vision—Techniques and future directions," IEEE Access, vol. 10, pp. 107293–107329, 2022, doi: 10.1109/ACCESS.2022.3209825.

[51] S. Zhang, C. Wang, S.-C. Chan, X. Wei, and C.-H. Ho, "New object detection, tracking, and recognition approaches for video surveillance over camera network," IEEE Sensors J., vol. 15, no. 5, pp. 2679–2691, May 2015, doi: 10.1109/JSEN.2014.2382174.

[52] S. Gothane, "A practice for object detection using YOLO algorithm," Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol., vol. 7, no. 2, pp. 268–272, 2021.

[53] S. Gupta et al., "Object detection with audio comments using YOLO v3," in Proc. Int. Conf. Appl. Artif. Intell. Comput. (ICAAIC), 2022.

[54] P. Druzhkov and V. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," Pattern Recognit. Image Anal., vol. 26, no. 1, pp. 9–17, 2016.

[55] H.-K. Jung and G.-S. Choi, "Improved YOLOv5: Efficient object detection using drone images under various conditions," Appl. Sci., vol. 12, no. 14, p. 7255, 2022.

[56] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified real-time object detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 779–788.

[57] A. Vidyavani, K. Dheeraj, M. R. M. Reddy, and K. N. Kumar, "Object detection method based on YOLOv3 using deep learning networks," Int. J. Innov. Technol. Explor. Eng. (IJITEE), vol. 9, no. 1, Nov. 2019.

[58] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint, arXiv:1804.02767, 2018.

[59] M. Kalinina and P. Nikolaev, "Research of YOLO architecture models in book detection," Adv. Intell. Syst. Res., vol. 174, 2019.

[60] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," Electronics, vol. 9, no. 3, p. 537, 2020.

[61] S. Gupta et al., "Object detection with audio comments using YOLO v3," in Proc. Int. Conf. Appl. Artif. Intell. Comput. (ICAAIC), 2022.

[62] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint, arXiv:1804.02767, 2018.

[63] A. Malta, M. Mendes, and T. Farinha, "Augmented reality maintenance assistant using YOLOv5," Appl. Sci., vol. 11, no. 11, p. 4758, 2021.

[64] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," Electronics, vol. 9, no. 3, p. 537, 2020.

[65] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified real-time object detection," arXiv preprint, arXiv:1506.02640, 2015.

[66] R. Sumalatha, R. V. Rao, and S. M. R. Devi, "Pothole detection using YOLOv2 object detection network and convolutional neural network," in Appl. Inf. Process. Syst., Singapore: Springer, 2022, pp. 293–300.

[67] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: A real-time object detection method for constrained environments," IEEE Access, vol. 8, pp. 1935–1944, 2019.

[68] Y. Du et al., "Pavement distress detection and classification based on YOLO network," Int. J. Pavement Eng., vol. 22, pp. 1659–1672, 2020.

[69] K. Li et al., "Object detection in optical remote sensing images: A survey and a new benchmark," ISPRS J. Photogramm. Remote Sens., vol. 159, pp. 296–307, 2020.

[70] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv preprint, arXiv:2004.10934, 2020.

[71] Q. Wu et al., "Object detection of double-sided copper laminates based on YOLOv5," in Proc. IEEE Int. Conf. Power, Electron. Comput. Appl. (ICPECA), Shenyang, China, 2023, pp. 171–175, doi: 10.1109/ICPECA56706.2023.10075704.

[72] X. Zhang et al., "Improvement of YOLOv5 model based on the structure of multiscale domain adaptive network for crowdscape," in Proc. IEEE Int. Conf. Cloud Comput. Intell. Syst. (CCIS), Xi'an, China, 2021, pp. 171–175, doi: 10.1109/CCIS53392.2021.9754600.

[73] A. Nithin and K. Jaisharma, "A deep learning-based novel approach for detection of face mask wearing using enhanced SSD over CNN with improved accuracy," in Proc. Int. Conf. Bus. Anal. Technol. Security (ICBATS), Dubai, UAE, 2022, pp. 1–5, doi: 10.1109/ICBATS54253.2022.9759018.

[74] D. Shyam, A. Kot, and C. Athalye, "Abandoned object detection using pixel-based finite state machine and single shot multibox detector," in Proc. IEEE Int. Conf. Multimedia Expo (ICME), San Diego, CA, USA, 2018, pp. 1–6, doi: 10.1109/ICME.2018.8486464.

[75] G. Zhang, S. Lu, and W. Zhang, "CAD-Net: A context-aware detection network for objects in remote sensing imagery," IEEE Trans. Geosci. Remote Sens., vol. 57, no. 12, pp. 10015–10024, 2019.

[76] J. Han et al., "ReDet: A rotation-equivariant detector for aerial object detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Nashville, TN, USA, 2021, pp. 2768–2795.

[77] X. Yang, J. Yang, J. Yan et al., "SCRDet: Towards more robust detection for small, cluttered and rotated objects," in Proc. IEEE/CVF Int. Conf. Comput. Vis., Seoul, Korea, 2019, pp. 8231–8240.

[78] J. Ding, N. Xue, Y. Long et al., "Learning ROI transformer for oriented object detection in aerial images," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Long Beach, CA, USA, 2019, pp. 2844–2853.

[79] J. Han, J. Ding, J. Li et al., "Align deep features for oriented object detection," IEEE Trans. Geosci. Remote Sens., vol. 60, pp. 1–11, 2021.

[80] X. Yang, J. Yan, Z. Feng et al., "R3Det: Refined single stage detector with feature refinement for rotating object," in Proc. 35th AAAI Conf. Artif. Intell., Virtual Event, 2021, pp. 3163–3171.

[81] X. Yang and J. Yan, "Arbitrary-oriented object detection with circular smooth label," in Proc. Eur. Conf. Comput. Vis., Cham: Springer, vol. 12353, 2020, pp. 677–694.

[82] G. S. Xia, X. Bai, J. Ding et al., "DOTA: A large-scale dataset for object detection in aerial images," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Salt Lake City, UT, USA, 2018, pp. 3974–3983.

[83] J.-A. Kim, J.-Y. Sung, and S.-H. Park, "Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition," in Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia), Seoul, Korea, 2020, pp. 1–4, doi: 10.1109/ICCE-Asia49877.2020.9277040.

[84] X. Yang, H. Sun, K. Fu et al., "Automatic ship detection in remote sensing images from Google Earth of complex scenes based on multiscale rotation dense feature pyramid networks," Remote Sens., vol. 10, no. 1, Art. no. 132, 2018.

[85] K. Fu, Z. Chang, Y. Zhang et al., "Rotation-aware and multi-scale convolutional neural network for object detection in remote sensing images," ISPRS J. Photogramm. Remote Sens., vol. 161, pp. 294–308, 2020.

[86] Z. Liu, H. Wang, L. Weng et al., "Ship rotated bounding box space for ship extraction from high-resolution optical satellite images with complex backgrounds," IEEE Geosci. Remote Sens. Lett., vol. 13, no. 8, pp. 1074–1078, 2016.

[87] S. Ren, K. He, R. Girshick et al., "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, 2017.

[88] L. Zhou, H. Wei, H. Li et al., "Objects detection for remote sensing images based on polar coordinates," arXiv preprint, arXiv:2001.02988, 2020.

[89] J. Yi, P. Wu, B. Liu et al., "Oriented object detection in aerial images with box boundary-aware vectors," in Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis., Waikoloa, HI, USA, 2021, pp. 2149–2158.

[90] W. Li, Y. Chen, K. Hu et al., "Oriented reppoints for aerial object detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., New Orleans, LA, USA, 2022, pp. 1829–1838.

[91] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional networks," arXiv preprint, arXiv:1505.00853, 2015.

[92] D. Misra, "Mish: A self-regularized non-monotonic activation function," arXiv preprint, arXiv:1908.08681, 2019.

[93] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "A large-scale hierarchical image database," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Miami, FL, USA, 2009, pp. 248–255.

[94] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. Int. Conf. Mach. Learn., Long Beach, CA, USA, 2019, pp. 6105–6114.

[95] N. Ma, X. Zhang, M. Liu, H. Hu, and J. Sun, "Activate or not: Learning customized activation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Nashville, TN, USA, 2021, pp. 8028–8038.

[96] S. Thokrairak, K. Thibuy, C. Fongsamut, and P. Jitngernmadan, "Optimal object classification model for embedded systems based on pre-trained models," in Proc. 25th Int. Comput. Sci. Eng. Conf. (ICSEC), Chiang Rai, Thailand, 2021, pp. 307–312, doi: 10.1109/ICSEC53205.2021.9684656.

[97] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 408–417.

[98] Z. Jiang, Y. Liu, C. Yang, J. Liu, P. Gao, Q. Zhang, S. Xiang, and C. Pan, "Learning where to focus for efficient video object detection," in Proc. Eur. Conf. Comput. Vis., Cham, Switzerland: Springer, 2020, pp. 18–34.

[99] L. Lin, H. Chen, H. Zhang, J. Liang, Y. Li, Y. Shan, and H. Wang, "Dual semantic fusion network for video object detection," in Proc. 28th ACM Int. Conf. Multimedia, Oct. 2020, pp. 1855–1863.

[100] L. Han, P. Wang, Z. Yin, F. Wang, and H. Li, "Exploiting better feature aggregation for video object detection," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), 2021.

[101] M. Han, Y. Wang, X. Chang, and Y. Qiao, "Mining inter-video proposal relations for video object detection," in Proc. Eur. Conf. Comput. Vis., Cham, Switzerland: Springer, 2020, pp. 431–446.

[102] G. Sun, Y. Hua, G. Hu, and N. Robertson, "MAMBA: Multi-level aggregation via memory bank for video object detection," in Proc. AAAI Conf. Artif. Intell., vol. 35, 2021, pp. 2620–2627.

[103] A. Vaswani et al., "Attention is all you need," in Proc. Int. Conf. Neural Inf. Process. Syst., 2017, pp. 5998–6008.

[104] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7794–7803.

[105] M. Zaheer et al., "Big bird: Transformers for longer sequences," in Proc. Int. Conf. Neural Inf. Process. Syst., 2020, pp. 1–15.

[106] R. Child, S. Gray, A. Radford, and I. Sutskever, "Generating long sequences with sparse transformers," arXiv preprint, arXiv:1904.10509, 2019.

[107] G. Zhao et al., "Explicit sparse transformer: Concentrated attention through explicit selection," arXiv preprint, arXiv:1912.11637, 2019.

[108] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?" in Proc. Int. Conf. Mach. Learn., 2021, pp. 813–824.

[109] B. Duke et al., "SSTVOS: Sparse spatiotemporal transformers for video object segmentation," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 5912–5921.

[110] X. Zhu et al., "Deformable DETR: Deformable transformers for end-to-end object detection," in Proc. Int. Conf. Learn. Represent., 2020, pp. 1–11.

[111] Q. Zhou et al., "TransVOD: End-to-end video object detection with spatial-temporal transformers," arXiv preprint, arXiv:2201.05047, 2022.

[112] 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.

[113] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 1492–1500.

[114] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in Proc. Int. Conf. Learn. Represent., 2016, pp. 1–10.

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

[116] 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, 2023.

[117] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified real-time object detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2016, pp. 779–788.

[118] A. Vidyavani, K. Dheeraj, M. R. M. Reddy, and K. N. Kumar, "Object detection method based on YOLOv3 using deep learning networks," Int. J. Innov. Technol. Explor. Eng. (IJITEE), vol. 9, no. 1, Nov. 2019.

[119] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint, arXiv:1804.02767, 2018.

[120] M. Kalinina and P. Nikolaev, "Research of YOLO architecture models in book detection," Adv. Intell. Syst. Res., vol. 174, 2019.

[121] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," Electronics, vol. 9, no. 3, p. 537, 2020.

[122] S. Gupta et al., "Object detection with audio comments using YOLOv3," in Proc. Int. Conf. Appl. Artif. Intell. Comput. (ICAAIC), 2022.

[123] D. T. Nguyen et al., "A high-throughput and power-efficient FPGA implementation of YOLO CNN for object detection," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 27, no. 8, pp. 1861–1873, Aug. 2019.

[124] F. U. D. Farrukh et al., "Power efficient tiny YOLO CNN using reduced hardware resources based on Booth multiplier

and WALLACE tree adders," IEEE Open J. Circuits Syst., vol. 1, pp. 76–87, 2020.

[125] C. Wang, Y. Zhou, and J. Li, "Lightweight YOLOv4 target detection algorithm fused with ECA mechanism," Processes, vol. 10, no. 7, p. 1285, 2022.

[126] R. Andri et al., "ChewBaccaNN: A flexible 223 TOPS/W BNN accelerator," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2021, pp. 1–5.

[127] I. Hubara et al., "Binarized neural networks," in Proc. Adv. Neural Inf. Process. Syst. (NIPS), vol. 29, 2016.

[128] C. Szegedy et al., "Going deeper with convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.

[129] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, Dec. 2015.

[130] H. Nakahara et al., "A lightweight YOLOv2: A binarized CNN with a parallel support vector regression for an FPGA," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays, Feb. 2018, pp. 31–40.

[131] T. B. Preuser et al., "Inference of quantized neural networks on heterogeneous all-programmable devices," in Proc. Design Autom. Test Eur. Conf. Exhib. (DATE), Mar. 2018, pp. 833–838.

[132] R. Gai, N. Chen, and H. Yuan, "A detection algorithm for cherry fruits based on the improved YOLO-v4 model," Neural Comput. Appl., vol. 35, no. 19, pp. 13895–13906, 2023.

[133] D. Dlužnevskij et al., "Investigation of YOLOv5 efficiency in iPhone-supported systems," Baltic J. Mod. Comput., vol. 9, 2021.

[134] S. Kim, S. Na, B. Y. Kong, J. Choi, and I.-C. Park, "Real-time SSDLite object detection on FPGA," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 29, no. 6, pp. 1192–1205, Jun. 2021.

[135] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 4510–4520.

[136] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 7263–7271.

[137] X. Chen, J. Xu, and Z. Yu, "A 68-mW 2.2 Tops/W low bit-width and multiplierless DCNN object detection processor for visually impaired people," IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 11, pp. 3444–3453, Nov. 2019.

[138] S. Yin et al., "A high energy efficient reconfigurable hybrid neural network processor for deep learning applications," IEEE J. Solid-State Circuits, vol. 53, no. 4, pp. 968–982, Dec. 2017.

[139] J.-S. Park et al., "A 6K-MAC feature-map-sparsity-aware neural processing unit in 5nm flagship mobile SoC," in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2021, pp. 152–154.

[140] A. Agrawal et al., "A 7nm 4-core AI chip with 25.6 TFLOPS hybrid FP8 training, 102.4 TOPS INT4 inference, and workload-aware throttling," in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, vol. 64, Feb. 2021, pp. 144–146.

[141] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 31, 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/9908279ebbf1f9b250ba689db6a0222b-Paper.pdf

[142] L. Liu et al., "On the variance of the adaptive learning rate and beyond," in Proc. Int. Conf. Learn. Represent. (ICLR), 2020. [Online]. Available: https://openreview.net/forum?id=rkgz2aEKDr

[143] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," arXiv preprint, arXiv:1608.03983, 2016.

[144] V. Sivakumar et al., "Comparison of object detection and patch-based classification deep learning models on mid- to late-season weed detection in UAV imagery," Remote Sens., vol. 12, p. 2136, 2020.

[145] 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, Sep. 2009.

[146] A. Paszke et al., "PyTorch: An imperative style high-performance deep learning library," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, Dec. 2019, pp. 8026–8037.

[147] L. Crockett, D. Northcote, C. Ramsay, F. Robinson, and R. Stewart, Exploring Zynq MPSoC: With PYNQ and Machine Learning Applications. Glasgow, Scotland: Strathclyde Academic Media, 2019.

[148] J. Lee et al., "UNPU: An energy-efficient deep neural network accelerator with fully variable weight bit precision," IEEE J. Solid-State Circuits, vol. 54, no. 1, pp. 173–185, Jan. 2019.

[149] Q. Huang et al., "CoDeNet: Efficient deployment of input-adaptive object detection on embedded FPGAs," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays, Feb. 2021, pp. 206–216.

[150] J. Bethge, C. Bartz, H. Yang, Y. Chen, and C. Meinel, "MeliusNet: An improved network architecture for binary neural networks," in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), Jan. 2021, pp. 1439–1448.

[151] S. P. K. Reddy and G. Kandasamy, "Cusp pixel labelling model for objects outline using R-CNN," IEEE Access, vol. 10, pp. 8883–8890, 2022, doi: 10.1109/ACCESS.2021.3139896.

[152] K. A. Hashmi et al., "Guided table structure recognition through anchor optimization," IEEE Access, vol. 9, pp. 113521–113534, 2021, doi: 10.1109/ACCESS.2021.3103413.

[153] W.-C. Lee, J.-Y. Zhang, and C.-C. Wei, "Using an LCD monitor and a robotic arm to quickly establish image datasets for object detection," IEEE Access, vol. 9, pp. 131006–131019, 2021, doi: 10.1109/ACCESS.2021.3111314.

[154] Z. Deng et al., "Multi-scale object detection in remote sensing imagery with convolutional neural networks," ISPRS J. Photogramm. Remote Sens., vol. 145, pp. 3–22, 2018.

[155] 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," arXiv preprint, arXiv:2207.02696, 2022.

[156] M. Ramesh, K. M. Rajpoot, and S. K. Rath, "A review on YOLOv8 and its advancements in object detection," Int. J. Comput. Vis. Image Process. (IJCVIP), vol. 14, no. 2, pp. 1–18, 2024.

[157] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. You, and P.-Y. Chen, "YOLOv8: A cutting-edge, real-time object detector with improved accuracy and speed," IEEE Trans. Image Process., 2024.