International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Exploring Object Detection Algorithms and implementation of YOLOv7 and YOLOv8 based model for weapon detection

Divya Kumawat[1], Dr. Deepak Abhayankar[2], Dr. Sanjay Tanwani[3]

Submitted: 25/01/2024 **Revised**: 03/03/2024 **Accepted**: 11/03/2024

Abstract: This paper explores the working principles, performance metrics, and architectural nuances of various object detection techniques, focusing mainly on the YOLO (You Only Look Once) algorithms. A comprehensive comparative study is conducted, considering factors such as loss function, backbone network, and performance on standardized image sizes. Beginning with an introduction, the paper classifies object detection algorithms, into one-stage and two-stage object detection techniques. The literature review scrutinizes the operational mechanisms and constraints of existing techniques. This study then transitions into weapon detection using the YOLOv7 and YOLOv8 algorithms, leveraging a dataset sourced and pre-processed from the Roboflow website. The mean Average Precision (mAP) achieved by YOLOv7 and YOLOv8 after training for 50 epochs stands at 0.9289 and 0.9430, respectively. Furthermore, the paper elucidates how performance metrics fluctuate with respect to epoch count in YOLOv7. In conclusion, the paper outlines avenues for further research, highlighting areas that warrant attention and exploration within the realm of object detection methodologies.

Keywords: *anchor box, classifier, feature map, RCNN, YOLO.*

1. Introduction

The surge in object detection's popularity is attributable to several key factors, including the ready availability of large annotated datasets, robust computational capabilities, a diverse array of network architectures, and the refinement of training methodologies. Object detection, which encompasses both object localization and classification, has become a cornerstone of modern computer vision systems. This work analyses different one-stage object detection and two-stage object detection techniques and their limitations. Classification of object detection algorithms:

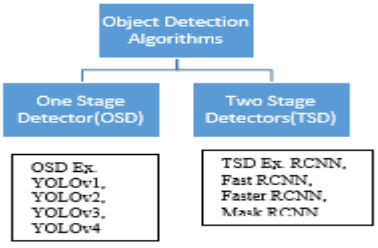


Fig.1. Classification of object detection techniques

OSD algorithms are also known as single-stage object detection techniques. They involve the prediction of both the bounding box and class in one pass through the network. The predictive model employed in this algorithm uses a grid-based approach to analyze images and make predictions. These algorithms are known for their high speed and suitability for real-time applications; however, they have false positives and small object localization errors.

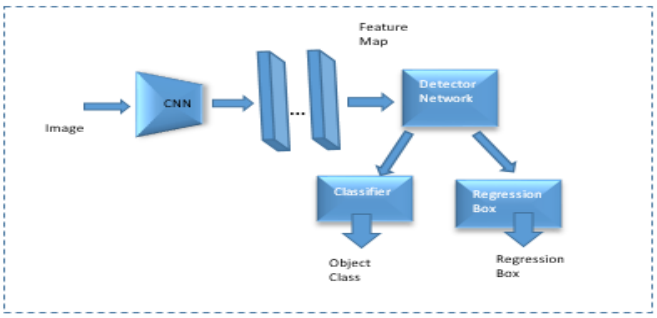


Fig. 2. Basic working of one stage object detector

In contrast, two-stage object detection methods operate in two distinct phases: an initial comprehensive scan of the entire scene followed by a focused analysis of regions of interest. These methodologies include a region proposal stage, which identifies potential object regions, and a refinement stage responsible for precise bounding box determination and class labelling. While two-stage algorithms may operate at a slower pace compared to OSD counterparts, they offer higher accuracy. The type of object detector required is decided on the basis of application needs, response time, and accuracy.

[11] *DAVV, Indore-452001, India*
ORCID ID : 0000-0003-1483-4332
[2] *DAVV, Indore-452001, India*
ORCID ID : 0000-0002-3328-1465
[3] *DAVV, Indore-452001, India*
ORCID ID :
** Corresponding Author Email: ddoraya@gmail.com*

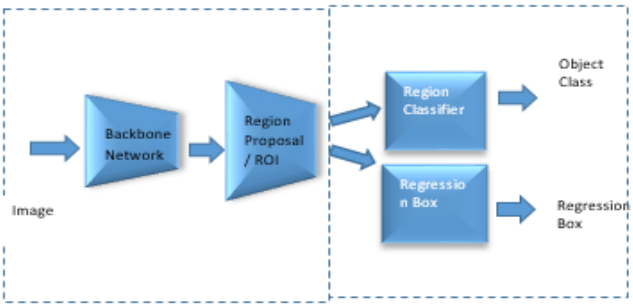


Fig. 3. Basic working of two-stage object detector

2. Literature Survey

Object detection algorithms have demonstrated a consistent ability to accurately detect and classify various real-world objects such as vehicles, persons, faces, buildings, and different objects in both images and videos. Object detection algorithms can be used in a variety of contexts, including human-computer interface (HCI), robotics, consumer electronics, transportation, security, and many more. Traditional techniques such as edge detection, template matching, and sliding window methods have given way to more advanced forms of object detection. Among the algorithms that use the traditional method are SIFT, which extracts scale-invariant, rotation-invariant, and transformation-invariant features[1]; Viola Jones, which uses haar-like features and a cascade classifier for face detection [2]; the Deformable Parts model, which considers an object as a deformable part and the relationship between them [3]; and the HOG algorithm using the distribution of gradients (HOG features) with SVM [4]. All of these algorithms have performance issues with complex images with many background details, occlusions, and scale variations. In 2012, Hinton et al. introduced the first CNN network named AlexNet in ILSVRC. It uses a graphical processing unit and a linear activation function to provide higher accuracy and reduce the error rate[5]. Consequently, new DCNNs are proposed yearly with some unique properties.

A region-based convolutional network by Girshik et al. revolutionized object detection in 2014. Its three components include a selective search algorithm-generated region proposal, a convolutional neural network that extracts features from input data and a class-specific SVM scores each feature vector. Multiple bounding boxes projected for a single object are often addressed with non-maximum suppression [6]. In addition to the RCNN algorithm, other widely recognized object detection algorithms within the RCNN family include Mask RCNN, Faster RCNN, and Fast RCNN. Fast RCNN provided by Girshik et al. improved accuracy and significantly increased speed over RCNN. The key improvements of Fast RCNN over RCNN are ROI pooling, shared convolutional features, and end-to-end training. Fast RCNN suffers from slow region proposal due to selective search on each grid cell, suboptimal performance due to discrepancies between training and inference, and low localization accuracy in small and overlapping objects [7]. Faster RCNN addresses these issues using region proposal network (RPN), unified training, and anchor boxes. In Faster RCNN, RPN and its other components can be trained together, resulting in alignment in training and inference. It uses an anchor box with a predefined set of reference bounding boxes for more accurate localization of objects [8]. Faster RCNN extension is Mask RCNN. It produces a bounding box and segmentation mask of each object for deep investigation of its borders and forms. Mask RCNNs have parallel branches with bounding box regression for mask prediction [9]. Table-I shows a comparison of the different RCNN family algorithms.

Many object detection methods uses Intersection over Union (IoU) threshold to predict an Object in an image. The threshold value signifies the quality of the findings. A threshold of 0.5 can produce low-quality results, whereas a high threshold value will impair the model's performance. To solve these issues, the Cascade RCNN was proposed in 2018. It uses a cascaded detector-trained incremental IoU threshold. Here, the output of one detector trains the next. It ensures a positive training set of equal size for all detectors and avoids overfitting[10].

The one-stage object detection technique features a streamlined detection pipeline and operates very instantaneously. Examples include various YOLO versions, SSDs, retinanet, and EfficientDet. Over the time, this techniques has improved its accuracy, efficiency, and precision while simultaneously increasing in complexity. A real-time object detector requires a faster and stronger architecture for feature detection, better feature integration capabilities, accurate detection, a stronger loss function, and better label assignment and training. Different versions of YOLO have been discussed below:

2.1. YOLOv1

YOLOv1 was proposed in 2016 using a deep CNN as the backbone. The image in YOLOv1 is partitioned into S*S grid cells. For a given class probability, two bounding boxes are produced using grid cells; one bounding box is chosen using the non-maximum suppression method. This enclosing bounding box is accompanied by a probability and class designation. An unstable gradient causes the model to diverge often when a high learning rate is employed. Dropout or significant data augmentation can be employed to prevent the overfitting issue [11]. YOLOv1 lags in detecting small objects and localization accuracy.

Fig. 4. Timeline of various RCNN family and YOLO family object learning algorithm

2.2. YOLOv2

YOLOv2 improves localization accuracy and small object recognition with anchor boxes and feature maps of different scales. YOLOv2 struggles to recognize very small, huge, and overlapping objects and performs computation-intensive training. It divides the image into a grid. YOLOv2 treats these grids independently and does not consider the relationship that may exist between objects in a different grid. One of its improvements, YOLO9000, has provided a 2% increase in mAP by including batch normalization. It trains the network on detection and classification simultaneously by using the COCO dataset for detection and the Imagenet dataset for classification. This joint training enabled YOLO9000 to train the network on a dataset that does not have labelled detection data [12]. It calculates the loss function on every anchor box and provides a high-resolution feature map. The major drawback of YOLOv2 was its inability to detect multi-scale images.

2.3. YOLOv3

YOLOv3 uses the concept of a feature pyramid, where an image is analyzed at three different scales and uses three detection heads to process different-sized feature maps and detect objects of different scales. It uses the multi-label concept, where the same object can be classified into different categories by using independent logistics classifiers and binary cross-entropy loss for class predictions [13].

2.4. YOLOv4

YOLOv4 has different components, namely the backbone, neck, and head. Backbone works as a features extractor by using one of the models, such as VGG16, shuffleNet, MobileNet, and squeezeNet. These features are passed to the neck, performing feature aggregation. And at last, the head is used for object detection. Backbone and detection both use the bag of freebies and the bag of specials. In order to overcome issues like small dataset size or diversity of data and increase training quality, Backbone uses the CutMix and Mosaic data augmentation techniques. Backbone also uses DropBlock regularization to explore different paths in the network and prevent overfitting, and class label smoothing to prevent the model from getting overconfident on predictions[14]. Backbone uses Mish activation (an alternative to YOLO that brings non-linearity to the network), cross-stage partial connections (connecting multiple stages of the network together for improved feature extraction capability), and multi-input weighted residuals (combining features from different scales and levels) as a bag of specials. The detector uses CIoU (Complete Intersection over Union), CmBN (Cross Mini-Batch Normalizations), self-adversarial training, and several anchors pointing to same ground truth, a cosine annealing scheduler to periodically reduce the learning rate, grid sensitivity elimination, and random training shapes as a bag of freebies. The detector has Mish activation, SPP-block (special pyramid pooling block), SAM block (Spatial Attention Module), PAN path, and Distance IoU [15]. The block diagram of the backbone, neck, and head with their respective algorithms is listed below:

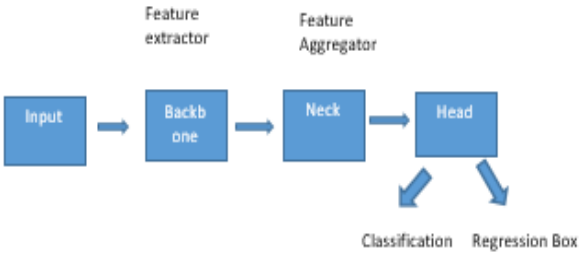


Fig. 5. Basic components of object detection algorithms used after YOLOv4

2.5. YOLOv5

In 2020, Ultralytics proposed YOLOv5 and made it open-source. It outperforms the earlier versions because of the dynamic architecture using the new CSP-Darknet 53 as a backbone, Spatial Pyramid Pooling Fast (SPPF), and CSP-PAN as a neck. SPPF accelerates computation by combining features into a map of fixed dimensions. It produced the final output using the same YOLOv3 head. To enhance the dataset's quality, it has included mosaic augmentation, copy-and-paste augmentation, arbitrary affine transformation (random rotation, scaling, shearing, and translation), mix-up augmentation, albumentation, and HSV augmentation. [16]. YOLOv5 has maintained a high level of speed while improving accuracy and efficiency. It employed an auto-anchor training technique to examine and test ill-fitted anchor boxes in the dataset [17].

2.6. YOLOv6

YOLOv6 was introduced in 2022 with two scalable, reparameterizable backbones and necks that can fit models of different sizes. For better speed and accuracy trade-offs, it uses the RepVGG reparameterization technique, which separates the multi-branch structure used during training time from the plain structure used during inference time. For training small models, Rep-Block (a collection of RepVGG blocks activated by ReLU) is used as the backbone, while CSPStackRep Block is used for large and medium-sized

networks. YOLOv6's neck component is built on a modified PAN topology derived from YOLOv4 and YOLOv5. It employs an efficient decoupled head that divides the classification and regression operations into two distinct branches. These modifications reduce computation costs further in order to decrease inference latency. The regression task predicts the regression box. A model based on anchor points can be used to predict the distance between the anchor point and the four sides of the bounding boxes. The model performed better when it was trained with the same optimizers and learning settings as YOLOv5 for 300 to 400 epochs [18]. Additionally, training for a longer time has been proven to yield improved results.

2.7. YOLOv7

In 2022, following a brief period of Yolov6, Yolov7 was released. It focuses on training process optimizations in addition to architecture optimization. To boost real-time object detection accuracy while keeping inference costs low, Wang et al. have developed a collection of trainable bag-of-freebies techniques. Model scaling is performed to achieve a good balance between aspects like the number of parameters used, accuracy, inference speed, and processing requirements. Model scaling can be achieved by using resolution, model depth, model channels, and the number of feature pyramids as scaling variables. Piotr Dollar et al. have analyzed individual scaling factors individually and also in a combined way. They came to this result by increasing the width of the models while scaling the depth and resolution to a lesser level [19]. The authors of YOLOv7 have addressed issues arising from the replacement of the original module by a re-parameterized module and output layer assignment by using a dynamic label assignment approach. They have developed the ideas of extended and compound scaling to make efficient use of both parameters and computational resources [20]. By controlling the shortest, highest-gradient path, deeper networks can train and converge more quickly. The major challenge is the gradient deviation. Expand, shuffle, and merge cardinality help E-ELAN improve network learning by controlling gradient path deviation. Fig. 6 shows ELAN's design. Group convolution increases the computational block channel and cardinality. Layered computational blocks share a group parameter and a channel multiplier. The feature maps generated by each processing unit are combined after being randomly partitioned into g-groups. Each cluster of feature maps will keep its previous configuration's number of channels. In the end, the feature map's g-group cardinality is added. While maintaining the ELAN design architecture, E-ELAN can be used to teach computational blocks new capabilities. The width of the computational block output grows when the depth of a model is scaled up. The result is a wider input to the gearbox layer. For this reason, YOLOv7 scales solely the depth of a computational block when using a model based on concatenation.

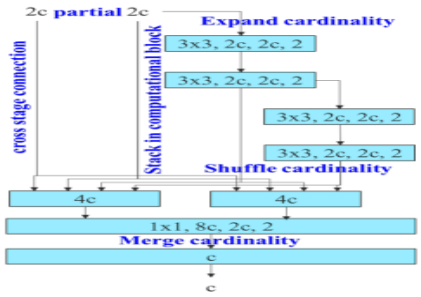


Fig. 6. Extended-ELAN[18]

2.8. YOLOv8

The YOLOv8 backbone is similar to YOLOv5, however, the C2f module replaces the CSP layer. Here, high-level characteristics and contextual information boost detection accuracy in the C2F module[21]. The decoupled head manages objectness, classification, and regression operations along with an anchor-free model. This layout improves model correctness and facilitates more focused work at each branch. The likelihood of an item being contained by a given bounding box is expressed by objectness probability. For objectness probability, a sigmoid activation function is used at the output layer. Class probabilities are represented by the softmax function. The model uses binary cross-entropy loss and CIoU, DFL loss for classification loss and bounding box loss respectively. These losses has specially increased small object detection [22].

In the Table 2 given below we are comparing YOLO algorithm from YOLOv1 to YOLOv8 on the basis of the architecture, image size, type of training used and performance.

YOLOv1 is known for its simplicity, but it does not incorporate the use of anchors. However, YOLOv2 and its subsequent versions have used the concept of anchors, thereby providing a noticeable improvement in accuracy. The use of anchors remained prevalent until the advent of YOLOX. The YOLOX onward version employed an anchor-free approach, which provides better accuracy than previous methods.

3. Comparison of Different Object Detection Techniques

3.1. Two-stage object detection techniques:

Table 1: Comparison of some core RCNN family algorithms

S. No.	Algorithm	Region proposal method used	mAP	Dataset Used for training	Drawbacks

1.	R-CNN [23]	Selective Search	58.5	Pascal VOC 2007	Network training is time-consuming as it classifies 2000 area proposals in each image.
2.	Fast-RCNN [7][12]	Selective Search	66.9	Pascal VOC 2007	Uses time consuming selective search region proposal generation algorithm
			70.0	Pascal VOC 2007+ 2012	
3.	Faster RCNN [8] [12]	Region proposal network	69.9	Pascal VOC 2007	It generates region proposal based on predefined anchor boxes which is not suitable for all type of objects
			73.2	Pascal VOC 2007+ 2012	

3.2. One-stage object detection techniques:

Table 2: Comparison of some core YOLO family algorithms

S. No.	Algorithm	i/p image size	Dataset Used	CNN Architecture used	Loss Function Used	Training technique	AP[24]
1	YOLO v1[11]	448*448	Pascal VOC	24 convolution layer with 2 fully connected layer	Multiple sum squared error(MSE)	Single stage training	63.4
2	YOLO 9000[12]	224*224 for 160 epoch and 448*448 for 10 epoch during training	Imagenet for training & Pascal voc	Darknet-19 with anchor box	MSE+cross entropy loss	Multi stage training	78.6 (on pascal voc 2007)s
3	YOLO v3[13]	416*416	MS-COCO	Darknet-53 with anchor box and SPP-Net	MSE+cross entropy loss+weighted smooth L1 loss	Multi Scale training	AP:36.2 AP50:60.6 At 20 FPS
4	YOLO v4[20]	Standard size for YOLOv4 is 416* 416 but this model has been trained on different image size	BDD 100K* MS COCO*	CSPDarknet53 with anchor box and FPN	MSE + cross entropy loss+ CIoU + GIoU loss	Multi Scale training with FPN and self adversial training	AP:43.5 AP50:65.7 At about 65 FPS
5	Scaled YOLO v4[25]	640*640	MS-COCO test dev 2017	scaled-down architecture(YOLOv4-Tiny) scaled-up model architecture (YOLOv4-Large)	MSE + cross entropy loss+ CIoU + GIoU loss	SGD Optimizer	AP:55.54 AP50:73.4 At 16 FPS [25]
6	YOLOR[26]		MS-COCO test dev 2017	YOLOv4-CSP	MSE + cross entropy loss+ CIoU + GIoU loss	Explicit and implicit knowledge based training	AP:55.4 AP50:73.3 At 30 FPS

7	YOLOX[27]	640*640	MS-COCO test dev 2017	YOLOv5 backbone with modified CSPNet, SiLU activation and PAN	BCE loss for objectness and classification. IoU for regression task	Training based on EMA weight updation, cosine lr schedule and IoU	AP:51.2 AP50:69.6 At 57.8 FPS
8	YOLOv5[16]	640*640	MS-COCO test dev 2017	CSPDarknet53 with anchor box and PAN	MSE + cross entropy loss + GIoU loss + DIoU loss + MSEoU loss[17]	Multi scale training with PAN, exponential moving average, mixed precision	AP:50.7
9	YOLO v6[18]	640*640	MS-COCO test dev 2017	EfficientRep with PAN	VariFocal Loss + SIoU/GIou*	Quantization aware training	AP:52.5 AP50:70 At 50 FPS
10	YOLOv7-E6[19]	1280*1280	MS COCO dataset test-dev 2017	E-ELAN	Assistant loss	Other trainable bag of freebies	AP:55.9 AP50:73.5 At 50 FPS
11	YOLO v8[24]	640 * 640	MS COCO dataset test-dev 2017	CSPDarknet53 with FPN and c2f module	CIoU, DFL and binary cross entropy	Adaptive training	AP:53.9 At 280 FPS

4. Weapon Detection by Using Yolov7 and Yolov8

The easy availability of CCTV cameras, thermal imaging cameras, and other sensor networks has paved the way for AI-enabled weapon detection. Object detection has gained popularity because of real-time object detection and increased accuracy. We have a large weapon dataset available, and a model can be trained to identify a weapon. This model can be used in many sectors and public places where enhanced security measures are required to meet security standards. In this paper, we are creating a model by using the YOLOv7 and YOLOv8 algorithms and comparing their performance.

4.1. Dataset Used

The weapon dataset used in this experimentation was acquired from the source https://roboflow.com/[28]. We have performed some preprocessing on this dataset. This weapon dataset has five different classes, namely: 0: Grenade, 1: Gun, 2: Knife, 3: Pistol, 4: Handgun, and 5: Rifle. The total images of these individual classes in train, test, and valid have been listed in the table III. Some of the sample train images are shown in Fig. 7.

TABLE 3: CLASS-WISE IMAGE COUNT IN DATASET

Class	Train images	Test Images	Valid images
0	3489	141	317
1	2882	109	253
2	2675	128	302
3	2908	102	246
4	893	38	73
5	1220	98	200
Total annotations	5829	274	612

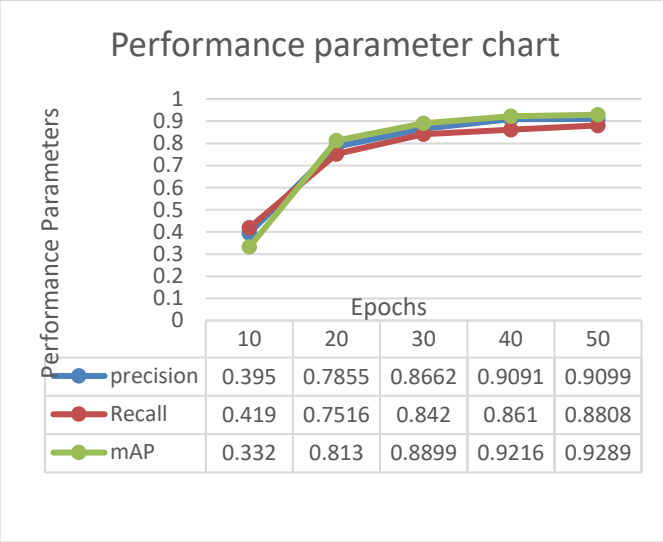
Fig. 7. Sample train images

4.2. Results & Discussion

The model's performance was evaluated through the measurement of mean Average Precision (mAP), recall, and precision at different epochs. Precision is an indicator that counts how many correctly identified objects there are out of all the known objects. Recall is the ratio of correctly identified objects to the known objects in an image. mAP assesses the comprehensive effectiveness of an object detection model across several categories and Intersection over Union (IoU) criteria. During YOLOv7 experiment, the model was trained using a dataset comprising 612 images that were accompanied by 1391 labels in each epoch. The

model underwent training across multiple epochs, and the evolution of performance parameters is presented in Table IV. It was observed that the performance parameter graph nearly reached a plateau after 50 epochs, with minimal or negligible changes to the performance parameter (as shown in fig of Table-IV).

Table 4: Illustration of the model precision, recall and mAP on different epochs



	10	20	30	40	50
precision	0.395	0.7855	0.8662	0.9091	0.9099
Recall	0.419	0.7516	0.842	0.861	0.8808
mAP	0.332	0.813	0.8899	0.9216	0.9289

The output obtained on some of the sample images is shown below in Fig 8. The model is providing precision of 0.9099 after 50 epochs. Furthermore, the system exhibits limitation of inability to identify multiple weapons in a bounding box. This problem can be solved up to an extent by increasing the dataset and number of epochs.

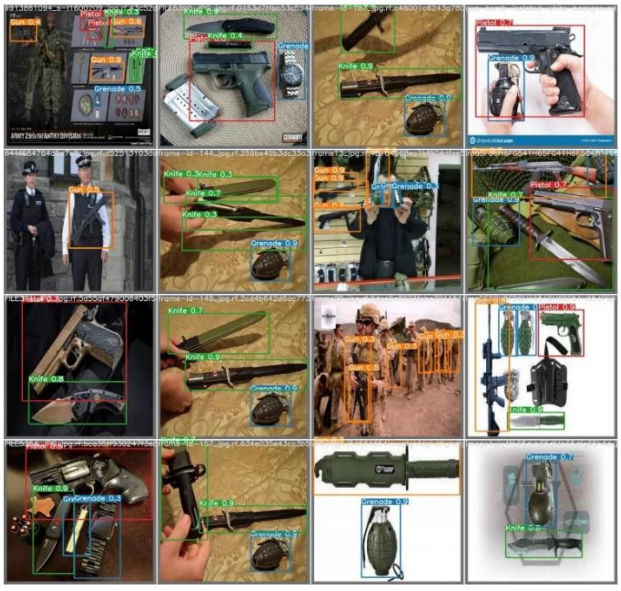


Fig.8. Models Predictions on test images

We have also measured the performance of our custom dataset on YOLOv8 and found that the performance of YOLOv8 is way better than YOLOv7. Performance of YOLOv8 is 0.02% better than YOLOv7 when model is trained on 50 epochs is shown in Table 5. YOLOv8 is giving better performance due to albumentation training which involves generation of more training dataset by applying blue, scaling augmentation techniques.

TABLE 5: YOLOV7 AND YOLOV8 PERFORMANCE ON WEAPON DATASET

S. No.	Algorithm Used	Precision	Recall	mAP
1	YOLOv7	0.9099	0.8808	0.9289
2	YOLOv8	0.922	0.897	0.943

5. Research Directions in Object Detection

Earlier object detection algorithms were working on information extracted from images such as edges, corners, and color histograms. These techniques are based on template matching or sliding windows and are time-consuming in nature. Subsequent to the year 2000, machine learning-based object detection has become more prevalent. In this paper, some cutting-edge algorithms are discussed, but there are still some unresolved concerns.

5.1. Detecting small objects:

Many advanced algorithms fail to identify small objects in an image. Increasing the image's quality or putting together high-resolution features can help in finding small objects. Mate Kisantal et al. have worked on small object augmentation on the MS-Coco dataset and achieved a 7.1% relative improvement in object detection [29].

5.2. Interpretability and transparency:

When an object detection algorithm is providing a prediction, its reasoning cannot be explained, and one is not certain about its prediction.

5.3. Real-time object detection:

These algorithms detect and identify objects with minimal delay in a resource-constrained environment. This delay time ranges between 20 and 50 ms for most of modern object detection techniques.

5.4. Robustness against domain shift:

Training and testing object detection algorithms on any dataset can provide good results, but when the model from the same algorithm and dataset is used in real time, it may not perform well. Mehran Khodabandeh et. al. have trained the model on the target domain by using noisy bounding boxes. These noisy bounding boxes are obtained from a trained detection model on the source domain [30].

5.5. Occlusion:

If one or more objects are blocking the view of an object, it can be difficult to identify that object. Antagtian Wang et. al. have used compositional nets for detecting occluded

objects. They have trained surrounding context-aware compositional nets by segmenting the surrounding context from the object. This algorithm has improved the relative performance of object detection by 41% [31].

5.6. Adversarial attack:

It is an intentional attack on data by introducing noise to it. This data seems normal to the human eye but leads to the misclassification of objects [32].

5.7. Lack of standard method:

There is no universal method to decide hyperparameter values such as number of layers, filter size, learning rate, etc. Further research is required in the field of hyperparameter optimization.

5.8. Quick training:

The object detection algorithm should be able to learn more features from a smaller training dataset as real-time data is limited. The size of training data can be increased by applying the augmentation techniques given in [33], but this will in turn increase the training time [34].

6. Conclusion

The evolution of computer vision has revolutionized various industries, healthcare, and security sectors, with object detection algorithms achieving real-time performance at the expense of increased complexity. The ongoing advancements in YOLO architecture, incorporating cutting-edge technologies such as deep learning and data augmentation, aim to enhance model performance and efficiency. However, significant challenges persist, including small object detection, occlusion handling, result interpretation, algorithm robustness across datasets, and susceptibility to adversarial attacks. As models improve, benchmark datasets such as COCO 2017 might be replaced by more challenging ones. YOLO's flexibility may lead to its application in diverse domains involving hard real-time response, like home appliances and driverless cars etc. This paper's analysis of YOLOv7 on a weapon dataset reveals that while performance improves with epoch count, challenges remain in identifying multiple weapons within an image. Our experiments achieved a maximum precision of 0.9099 in YOLOv7 and 0.922 in YOLOv8. Further investigation could involve exploring alternative YOLO algorithms, increasing epoch counts, and expanding dataset sizes to refine precision and address existing limitations.

7. Future Work:

Future research endeavors will extend the experimentation to evaluate alternative YOLO variants and explore other object detection algorithms. Additionally, increasing the epoch count and dataset size will provide valuable insights into scalability and robustness. Transfer learning and ensemble approaches will be used to enhance model performance and address current challenges in object detection.

Author contributions

Divya Kumawat1: Conceptualization, Methodology, Software, Field study **Dr. Deepak Abhayankar2:** Original draft preparation, Validation. Field study **Dr. Sanjay Tanwani3:** Visualization, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] D. G. Lowe, “Object recognition from local scale-invariant features,” in Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, pp. 1150–1157 vol.2. doi: 10.1109/ICCV.1999.790410.

[2] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2001, pp. I–I. doi: 10.1109/CVPR.2001.990517.

[3] P. Felzenszwalb, D. McAllester, and D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” in 2008 IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8. doi: 10.1109/CVPR.2008.4587597.

[4] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), 2005, pp. 886–893 vol. 1. doi: 10.1109/CVPR.2005.177.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in Advances in Neural Information Processing Systems, F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-Based Convolutional Networks for Accurate Object Detection and Segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[7] R. Girshick, “Fast R-CNN,” Proc. IEEE Int. Conf. Comput. Vis., vol. 2015 Inter, pp. 1440–1448, 2015, doi: 10.1109/ICCV.2015.169.

[8] 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, doi: 10.1109/TPAMI.2016.2577031.

[9] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 2, pp. 386–397, Feb. 2020, doi: 10.1109/TPAMI.2018.2844175.

[10] 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, 2021, doi: 10.1109/TPAMI.2019.2956516.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[12] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.

[13] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018.

[14] A. Bochkovskiy, C.-Y. Wang, and H. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020.

[15] C. Wang, A. Bochkovskiy, and H. M. Liao, "Scaled-YOLOv4 : Scaling Cross Stage Partial Network," pp. 13029–13038.

[16] "Ultralytics YOLOv5 Architecture," 2023. https://docs.ultralytics.com/yolov5/tutorials/architecture_description/#1-model-structure (accessed Jul. 19, 2023).

[17] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," AAAI 2020 - 34th AAAI Conf. Artif. Intell., no. 2, pp. 12993–13000, 2020, doi: 10.1609/aaai.v34i07.6999.

[18] C. Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications".

[19] C.-Y. Wang, A. Bochkovskiy, and H. Liao, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. 2022. doi: 10.48550/arXiv.2207.02696.

[20] J. Woo, J. Baek, S. Jo, and S. Y. Kim, "A Study on Object Detection Performance of YOLOv4 for Autonomous Driving of Tram," 2022.

[21] J. Terven, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," Mach. Learn. Knowl. Extr., vol. 5, no. 4, pp. 1680–1716, 2023, doi: 10.3390/make5040083.

[22] H. Lou et al., "DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor," Electron., vol. 12, no. 10, 2023, doi: 10.3390/electronics12102323.

[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[24] J. Terven and D. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," pp. 1–33, 2023, [Online]. Available: http://arxiv.org/abs/2304.00501

[25] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling Cross Stage Partial Network," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 13024–13033. doi: 10.1109/CVPR46437.2021.01283.

[26] C.-Y. Wang, I.-H. Yeh, and H. Liao, "You Only Learn One Representation: Unified Network for Multiple Tasks," J. Inf. Sci. Eng., vol. 39, pp. 691–709, 2021.

[27] Z. Ge, "YOLOX: Exceeding YOLO Series in 2021," pp. 1–7, 2021.

[28] Testing, "guns Computer Vision Project," Roboflow, 2022. https://universe.roboflow.com/testing-kfsrv/guns-l4rap

[29] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," Feb. 2019.

[30] M. Khodabandeh, A. Vahdat, M. Ranjbar, and W. G. Macready, "A Robust Learning Approach to Domain Adaptive Object Detection," Apr. 2019.

[31] A. Wang, Y. Sun, A. Kortylewski, and A. Yuille, "Robust Object Detection Under Occlusion With Context-Aware CompositionalNets," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Jun. 2020, pp. 12642–12651. doi: 10.1109/CVPR42600.2020.01266.

[32] A. Serban and E. Poll, "Adversarial Examples on Object Recognition :," vol. 53, no. 3, 2020, doi: 10.1145/3398394.

[33] [33] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep

Learning," J. Big Data, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0197-0.

[34] E. Arulprakash and M. Aruldoss, "A study on generic object detection with emphasis on future research directions," J. King Saud Univ. - Comput. Inf. Sci., vol. 34, no. 9, pp. 7347–7365, 2022, doi: 10.1016/j.jksuci.2021.08.001.