YOLOV5, YOLO-X, YOLO-R, YOLOV7 PERFORMANCE COMPARISON: A SURVEY

Ismat Saira Gillani[1], Muhammad Rizwan Munawar[2], Muhammad Talha[3], Salman Azhar[4], Yousra Mashkoor[5], Muhammad Sami uddin[6] and Usama Zafar[7]

[1]Department of Computer Science, Columbus State University, USA
[2]Department of Computer Science, COMSAT University Pakistan
[3]Department of Electrical Engineering, GC University Faisalabad, Pakistan
[4]Department of Creative Technology, Air University Pakistan
[5]Department of Computer Science, NED University Pakistan
[6]National University of Technology (NUTECH) Islamabad, Pakistan
[7]Department of Computer Science, GC University Faisalabad, Pakistan

ABSTRACT

YOLOv7 algorithm have taken the object detection domain by the storm as its real-time object detection capabilities out ran all other previous algorithms both in accuracy and speed [1]. *YOLOv7 advances the state of the art results in object detection by inferring more quickly and accurately than its contemporaries. In this paper, we are going to present our work of implementing this SOTA deep learning model on a soccer game play video to detect the players and football. As the result, it detected the players, football and their movement in real time. We also analyzed and compared the YOLOv7 results against its previous versions including YOLOv4, YOLOv5 and YOLO-R. The code is available at: https://github.com/RizwanMunawar/YOLO-RX57-FPS-Comparision*

KEYWORDS:

Deep Learning, Object detection, Image segmentation, Instance segmentation, Network Architecture.

1. INTRODUCTION

Real-time prediction of the presence of one or more objects, along with their classes and bounding boxes, is the task of computer vision that has taken the industry by storm. Object detection can use a neural network to classify and localize an object in the image. Benefitting from this capability, there is a tremendous amount of work that is being done in the different streams of life from facial recognition to autonomous driving cars, security applications and robotics [2]. Modern detectors have been in the development to identify the objects in higher frame rate. Recently, it has been seen how vehicles candrive on their own once put on the auto-pilot mode. However, their applications are not limited to these fields only. In this paper, state of the art YOLOv7 model will be utilized for real-time detection of the players and football movement. The results of this implementation were quite fascinating as there was highest accuracy seen along with speed. Generally, object detection methods are of two types; **a.** single-stage detectors, **b.** multi-stage detectors (Figure 1). The former methods directly deal with the execution speed whereas the latter focus more on the accuracy of the model and are evaluated in

MAP metric. Before YOLO, two stage detectors *(R-CNN, Fast R-CNN, and Faster R-CNN)* demonstrated the SOTA results in terms of accuracy [3].

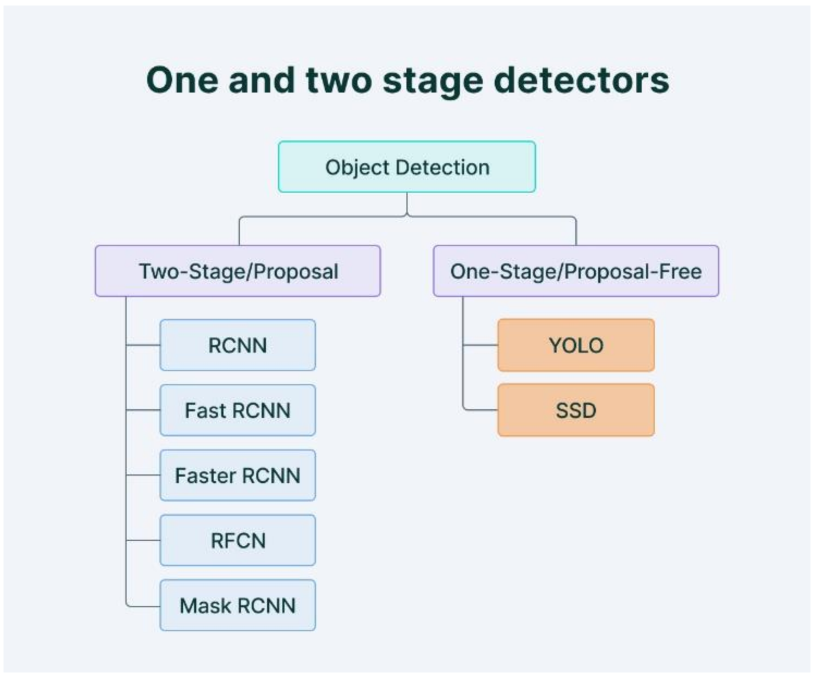

Figure 1. Comparison between One-Stage and Two-Stage Detectors

However, there is always a trade-off between speed and accuracy among these methods. When applied to real-time data, RCNN showed better accuracy as compared to RFCN which yielded more speed. Later on, YOLO model replaced other SOTA algorithms because of its speed and accuracy. YOLOv2 achieved results with a reasonable 78.6% *MAP* on *VOC 2007+2012* at 40 FPS [4]. Nevertheless, over the years many YOLO versions were developed and in each version there was a speed accuracy trade off.

Moving forward, the YOLOv5, YOLO-R and YOLOv7 will be talked about respectively. Firstly, the limitations of the first two models will be outlined and then improvements and structure of YOLOv7 will be discussed. Afterwards, we will compare the performance of all three models to analyze which one is the most accurate.

2. Related Work

The YOLO algorithm uses convolutional neural networks (CNN) to quickly identify objects. The approach just needs one forward propagation through a neural network, as the name would imply, to detect objects.

2.1. YOLOv5

For feature extraction from photos made up, it uses CSPDarknet as the foundation. In order to aggregate the features and pass them on to Head for prediction, it creates a feature pyramids network using PANet. Intuitively, the data is fed to the Backbone, which is nothing but aCSPDarknet, where feature extraction is happened and then they are passed to the Neck, aPANet, where the feature fusion takes place and then lastly, YOLO layer predicts the detection output in terms of class, location, confidence score and size [5]. *(Figure 3)*

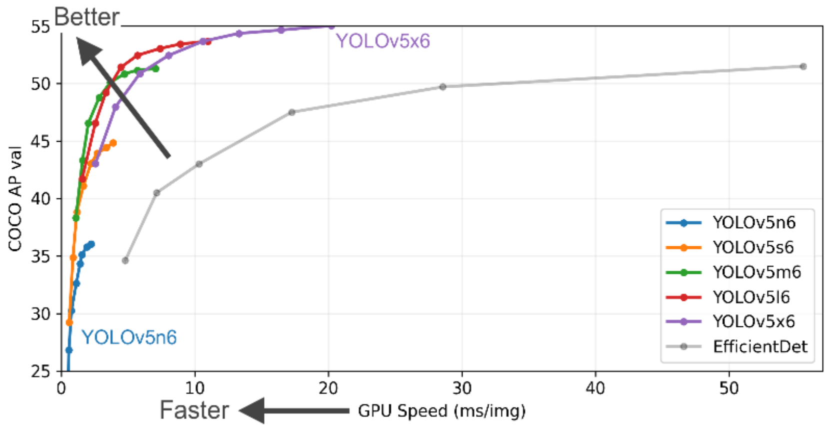

Figure 2. Comparison of Different YOLOv5 Variants

During training of this model, Leaky ReLU, sigmoid activation, SGD, and ADAM are available as optimizer choices in YOLOv5. It makes use of Logits loss and Binary cross-entropy. Just like every other model, it also has multiple variants and these varieties have a size and inference time trade off. The lightest variant YOLOv5s takes up to only 14MB however it lacks seriously in accuracy as compared to the YOLOv5x that has size of 168MB but is the most accurate in this family. It can be seen that YOLOv5x shows *48% AP* at 5ms per image as compared to YOLO5s that demonstrates *45% AP* at 5ms per image (*Figure 2)*.

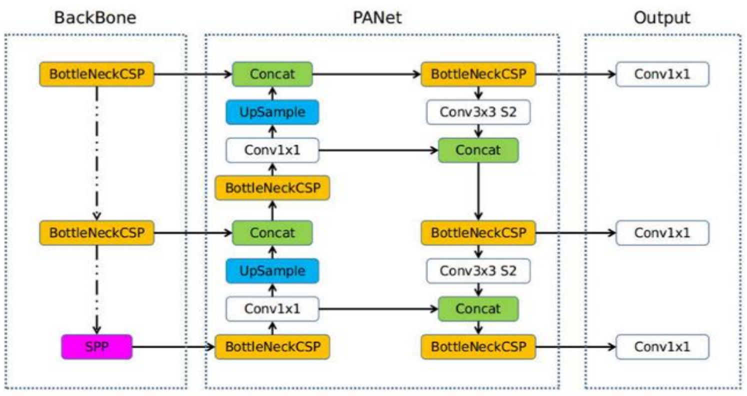

Figure 3. Network Architecture of YOLOv5 *[5]*

2.2. YOLO-R

This variant had the best inference time and accuracy among all YOLO models including YOLOv5. It proposes a single neural network that does multiple tasks like prediction, multi-tasking learning and feature calibration. YOLOR makes the most of Explicit and Implicit knowledge to build up the model that performs multi-label classification, detection, feature embedding all at once. This model learns from both the given data and the input (explicit knowledge) and the data learnt from the past experiences (implicit knowledge) [6].

A total of three processes that include kernel space alignment, prediction refinement and Convolutional Neural Network (CNN) make this architecture functional. CNNs try to retrieve an output according to an input. However, YOLOR yields both the CNNs that learn how to extract the output and all the possibilities of different outputs that could be, instead of just one result *(Figure 4)*.

When compared with YOLOv4, YOLOR-E6 delivers the better AP value with *5%* less computation. It use *10%* less parameters and improved inference speed by *15%*. In case of U5R5-X6, YOLOR demonstrates best AP with 29% less computation with *11%* parameters *(Table 1)*.

Table 1. Comparison between YOLOv4 and YOLOR [7]

Model	**Size**	**FPS**$^{TitanRTX}_{batch32}$	**FLOPs**	**# parameters**	**AP**^{val}	**AP**$^{val}_{50}$	**AP**$^{val}_{75}$	**AP**$^{val}_{S}$	**AP**$^{val}_{M}$	**AP**$^{val}_{L}$
YOLOR-P6	1280	72	326.2G	37265016	52.5%	70.6%	57.4%	37.4%	57.3%	65.2%
YOLOR-W6	1280	66	454.0G	79873400	54.0%	72.1%	59.1%	38.1%	58.8%	67.0%
YOLOR-E6	1280	39	684.0G	115909400	54.6%	72.5%	59.8%	39.9%	59.0%	67.9%
YOLOR-D6	1280	31	936.8G	151782680	55.4%	73.5%	60.6%	40.4%	60.1%	68.7%
Y4-P6	1280	34	718.4G	127530352	54.4%	72.7%	59.5%	39.5%	58.9%	67.3%
U5R5-S6	1280	139	69.6G	12653596	43.3%	61.9%	47.7%	29.0%	48.0%	53.3%
U5R5-M6	1280	93	209.6G	35889612	50.5%	68.7%	55.2%	35.5%	55.2%	62.0%
U5R5-L6	1280	67	470.8G	77218620	53.4%	71.1%	58.3%	38.2%	58.4%	65.7%
U5R5-X6	1280	36	891.6G	141755500	54.4%	72.0%	59.1%	40.1%	59.0%	67.2%

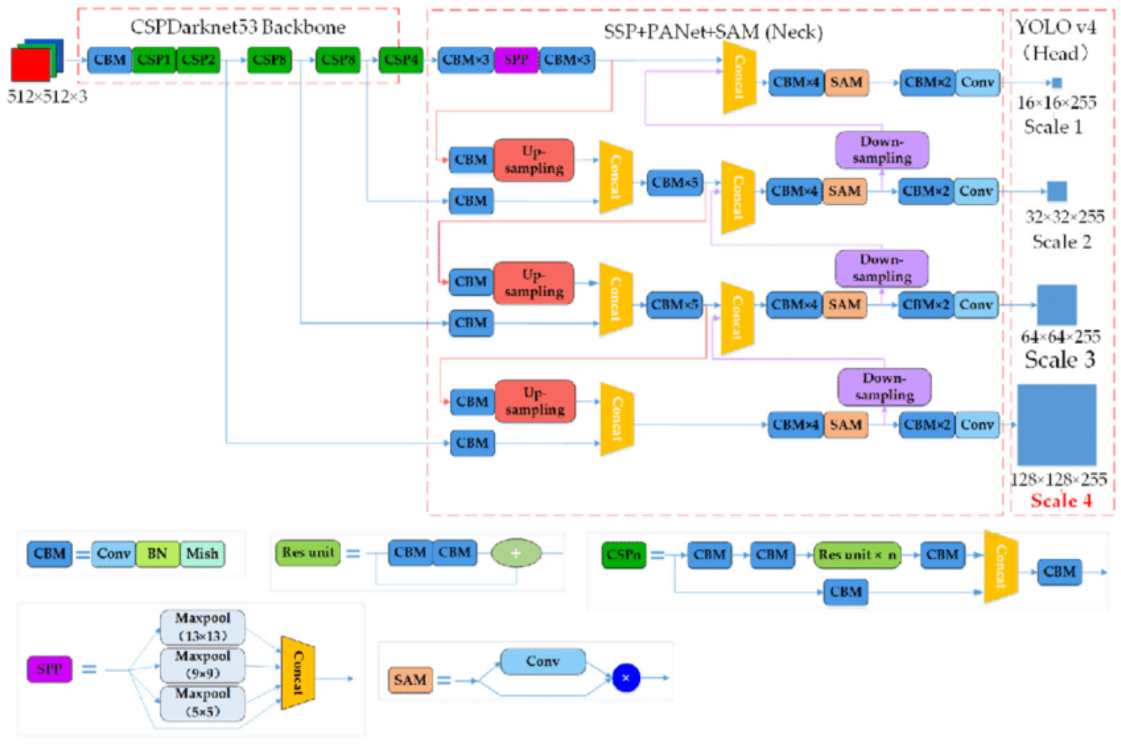

Figure 4. YOLO-R Architecture *[6]*

2.3. YOLO-X

After YOLOv5, advanced label assignment and anchor-free detectors grabbed the spotlight due to their better results. However, the YOLO family wasn't still integrated with these improvements which became the ultimate reason for the development of YOLOX model.

With a DarkNet53 backbone, YOLOX is a single-stage object detector that makes a number of changes to YOLOv3. In particular, YOLO's head is replaced with a decoupled one. Prior to adding two parallel branches with two 3 x 3 conv layers each for the classification and regression tasks, we adopt a 1 x 1 conv layer for each level of FPN features to minimize the feature channel to 256 [8].

To solve the conflict between classification and regression in object detection, this model uses Decoupled head architecture which means that classification and localization processes on the images are separated. This decoupled head architecture can be implemented on both regression and classification based models, one-stage and two-stage respectively.

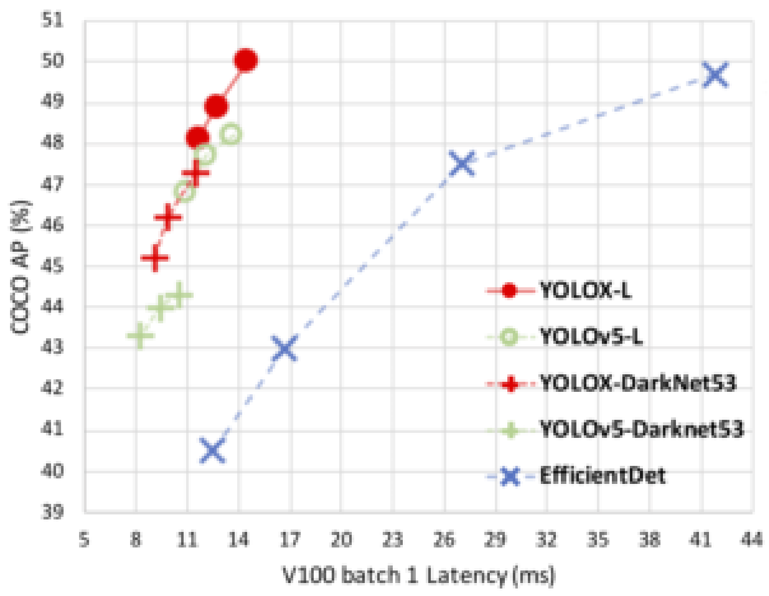

Figure 5. Comparison between YOLOX and YOLOv5 *[8]*

3. Method Used

In this paper, the YOLOv7 trained on MS COCO model was utilized on a soccer game play, detecting all the players in real-time and football movement. The inference time and accuracy of the model was carefully observed and then was compared with other models.

3.1. Major Improvements in Yolov7

YOLOv7 is the latest state of the art detector to this date among all known object detectors. Few architectural reforms have been introduced to the model that have improved the speed and the accuracy significantly. Just similar to scaled YOLOv4, this model also uses COCO dataset. The architectural reforms include E-ELAN (Extended Efficient Layer Aggregation Network) along with Model Scaling for concatenation based Models to enhance learning ability. Additionally, planned re-parameterized convolution and coarse for auxiliary with Fine for lead loss are aliased as the bag of freebies that also compliment the model for better learning without actually increasing the training cost [1].

Expand, shuffle, and merge cardinality is used in the proposed E-ELAN to continuously improve the network's capacity for learning while preserving the original gradient path. This block in YOLOv7 is based on the previously developed ELAN computational block *(Figure 6).*

Compound model scaling is another reason for the improved performance of YOLOv7. To make it more computing devices friendly, the model scaling was used to find the accuracy and speed requirements.

Usually for model scaling, *NAS* (Network Architecture Search) is used because of the parameter-specific scaling. However, in YOLOv7, the model is further optimized with a compound model scaling approach. In this approach, with and depth are coherently scaled for concatenation-based models *(Figure 7)*.

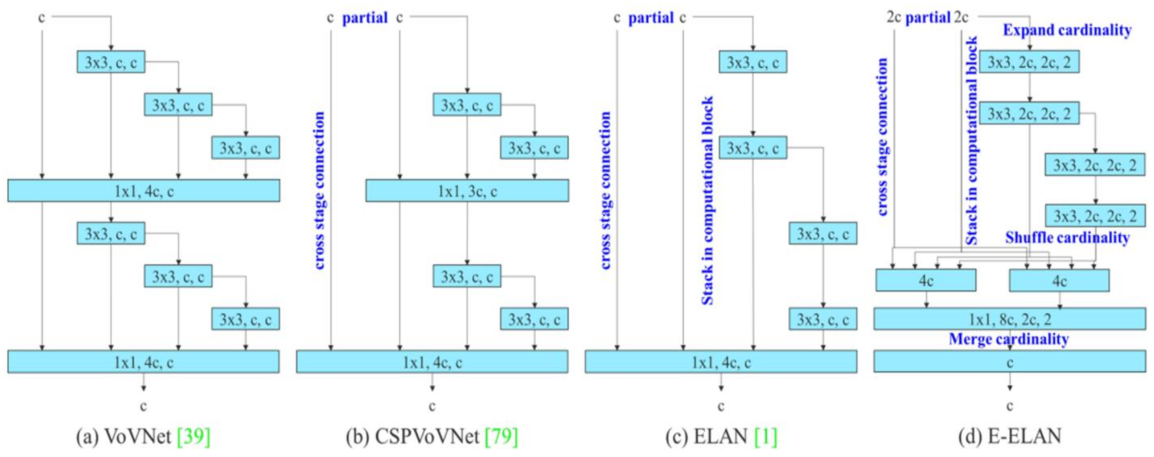

Figure 6. Comparison between VoVNET *[9]*, CSPVoVNET *[10]*, LAN and ELAN for better Network Learning

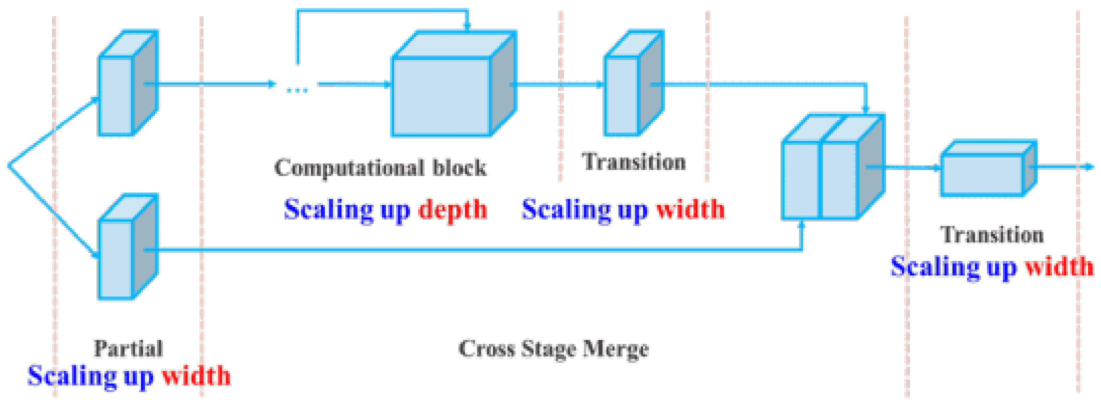

Figure 7. Compound Model Scaling in YOLOv7 *[1]*

Additionally, this model put forward the planned re-parameterization convolution that affects the accuracy significantly. Re-parameterization techniques [[11], [12], [13], [14], [15], [16]] are the method for enhancing the model after training. It lengthens the training process but yields better inference outcomes. Both Model level and Module level ensemble re-parametrization are the two types of re-parametrization used to finalize this models.

Model level re-parametrization is usually done either by training the multiple models with same settings but different training data and then taking the average of their weights or taking the average of the weights of models at multiple epochs. Recently, research on module level re-parameterization has exploded. This approach divides the model training process into a number of modules. To create the final model, the outputs are ensemble. *(Figure 8)*

The YOLO architecture usually contains a neck, a head and backbone and the outputs are obtained in the head. YOLOv7 has made few changes here as well. It is not constrained by a single head. It contains multiple heads to achieve whatever it wants. Earlier this method has also been used in Deep Supervision technique that is used by DL models to utilize multi heads.

There are two heads that. Firstly, the lead head in YOLOv7 is referred to as the final output head and secondly, the Auxiliary Head is the head that helps with middle-layer training. With the assistance of am assistant loss, auxiliary heads' weights are updated that allows the Deep Supervision [17] and the model's learning ability gets uplifted. This approach is closely related to Lead Head and the Label Assigner. Label Assigner is a method that assigns soft labels after taking the ground truth and network prediction outcomes into account. It's vital to notice that the label assigner creates coarse and soft labels rather than hard labels *(Figure 9)*.

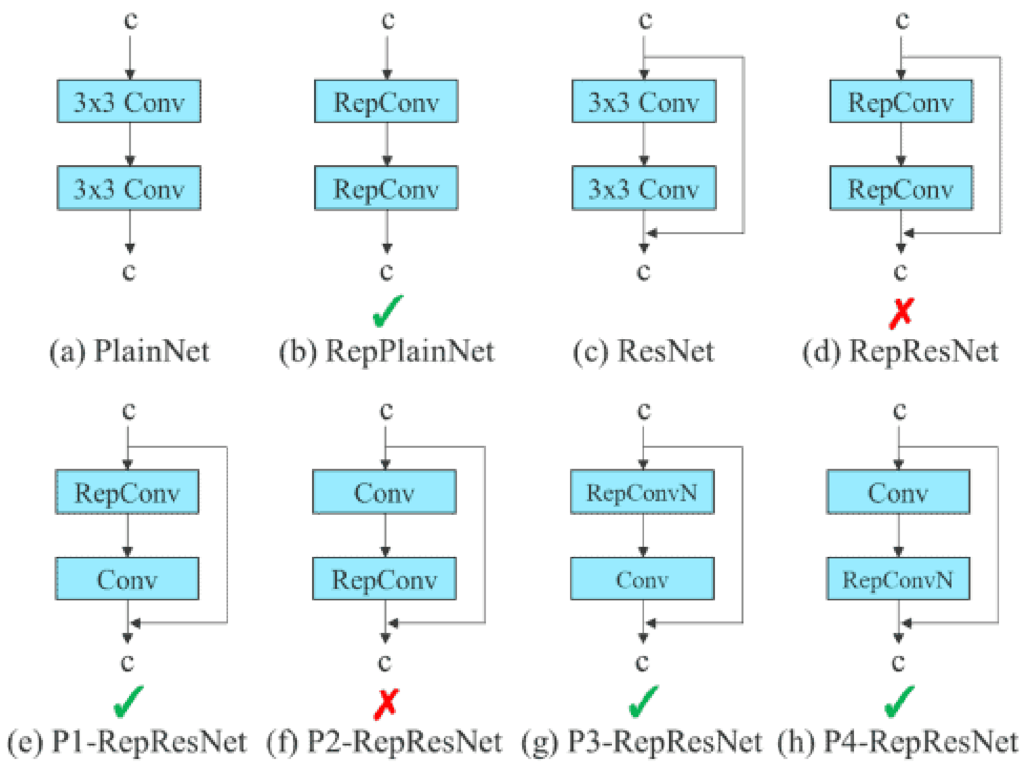

Figure 8. Best ways to perform module-level ensemble *[18]*

The YOLOv7 network's Lead Head makes predictions about the outcome. These final results are used to generate soft labels. The crucial aspect is that the identical soft labels that are generated are used to calculate the loss for both the lead head and the auxiliary head. In the end, the soft labels are used to train both heads. This is necessary because the lead head has a somewhat robust learning potential, which means that the soft label that results from it should be more accurate in capturing the distribution and correlation between the source and target data. The lead head will be better able to concentrate on learning residual information that has not yet been learnt by enabling the shallower auxiliary head immediately study the information that the lead head has learned.

Getting to the labels that go from coarse to fine *(Figure 9)*. Actually, two distinct soft labels sets are produced in the technique outlined above. The first one is fine label set for the lead head to train and a set of coarse labels for the auxiliary head's training. The fine labels are identical to the soft labels that were created immediately. More grids are, however, handled as positive targets in order to construct the coarse labels. To do this, the positive sample assignment procedure's limitations are relaxed.

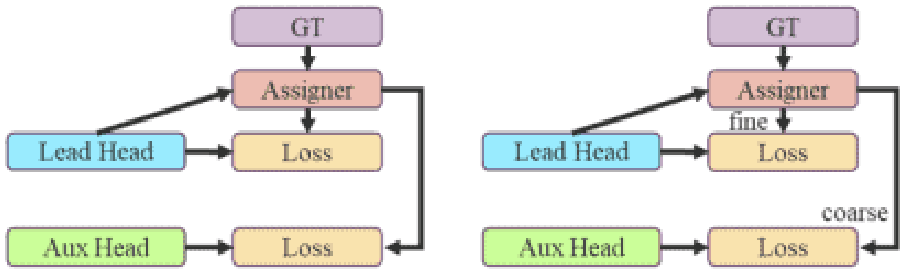

Figure 9. Lead guided Assigner and Coarse to fine Lead Guided Assigner

YOLOv7 demonstrates *56.8%* AP against *28 ms* inference time as compared to YOLO-R that exhibits *56.4%* AP against same inference time. All other models including YOLOv5, Scaled YOLOv4, PPYOLOE and YOLOX. All these models demonstrate lower *AP* as compared to the new YOLOv7 *(Figure 10)*.

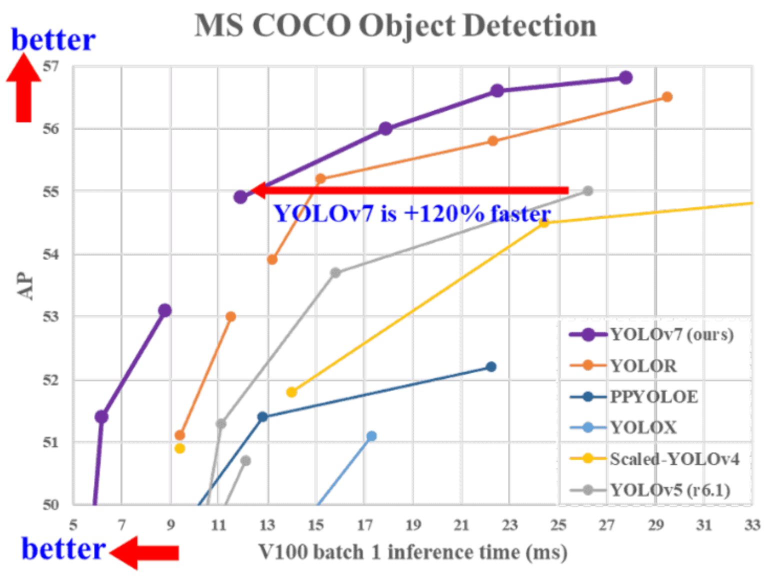

Figure 10. Comparison between YOLOv7 and other models of YOLO Family [1]

4. RESULTS AND COMPARISON BETWEEN YOLOV5, YOLO-X, YOLO-R AND YOLOV7

After applying YOLOv7 along with YOLOR, YOLOv5 and YOLOX on the data, the performance of all these models in terms of *FPS* and *accuracy* was examined.

4.1. Fps Comparison

When *YOLOv5m* was applied to the data it showed the speed on *27* Frames per second which is followed by *YOLOR-p6* that yielded in *23 FPS. YOLOXm* also gave the *FPS* of 22 that is slightly less than that of *YOLOR-p6*. Finally, lowest speed was displayed by *YOLOv7* that gives the *FPS* of *17* only. *(Figure 11)* Indeed, YOLOv5m is delivering the highest FPS rate.

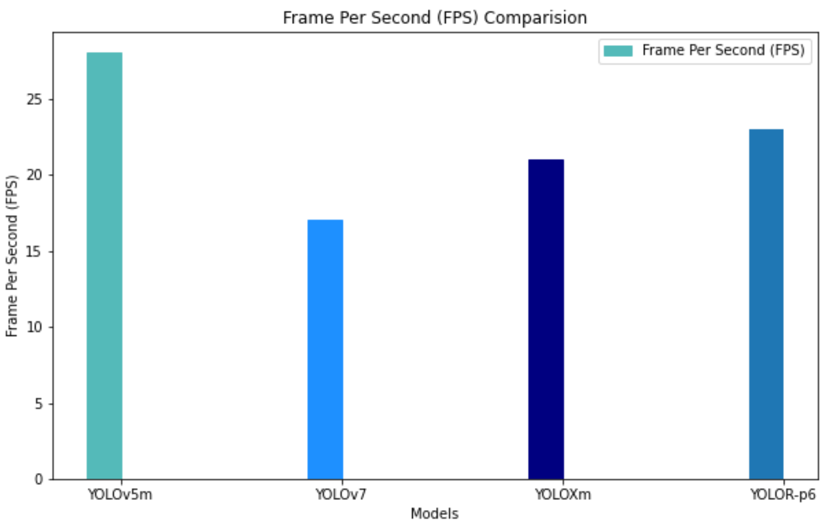

Figure 11. FPS Comparison between YOLOv5m, YOLO-p6, YOLOXm and YOLOv7

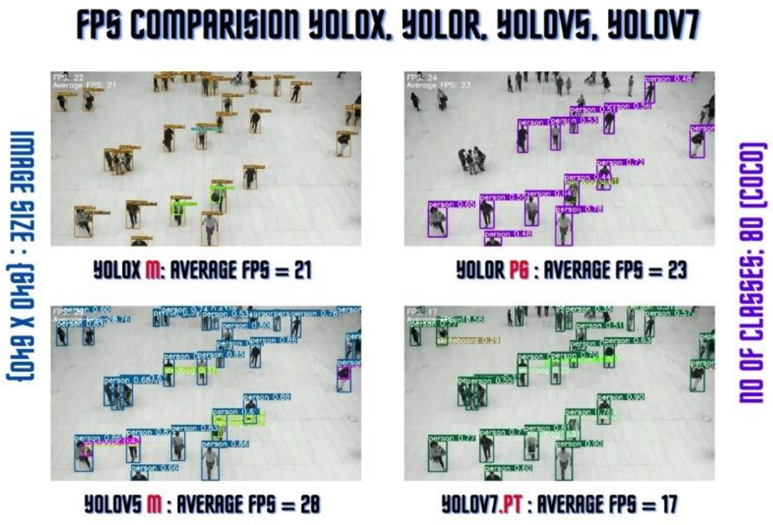
FPS COMPARISION YOLOX, YOLOR, YOLOV5, YOLOV7
IMAGE SIZE : (640 X 640)
NO OF CLASSES: 80 (COCO)
YOLOX M: AVERAGE FPS = 21
YOLOR P6 : AVERAGE FPS = 23
YOLOV5 M : AVERAGE FPS = 28
YOLOV7.PT : AVERAGE FPS = 17

Figure 12. FPS Comparison

4.2. Accuracy Comparison

The accuracy results of all these models were also compared and it was seen that even with the least speed, *YOLOv7* is delivering close to state of the art accuracy. Highest accuracy of *YOLORv7* can be seen in the metrics which is closely followed by the *YOLO-p6* with the accuracy of *56% AP*. Then comes the *YOLOv5m* with the accuracy of *53% AP* along with *YOLOXm* that gave the *46% AP (Figure 13)*.

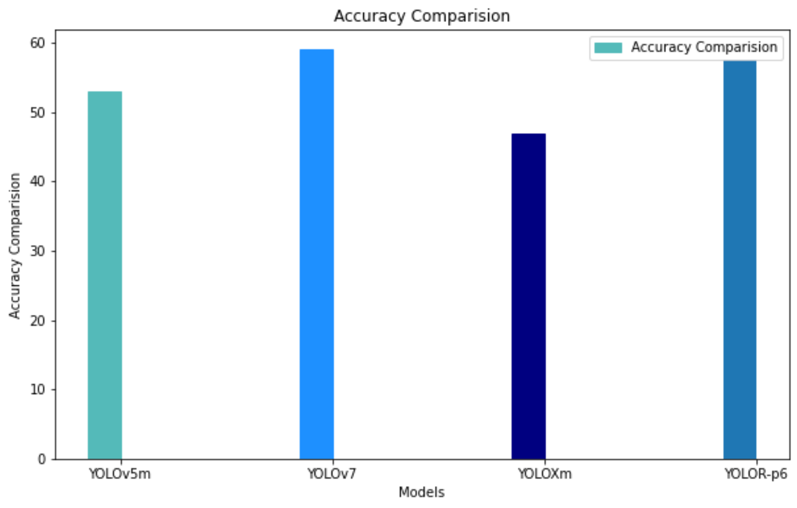

Figure 13. Accuracy Comparison between YOLOv5m, YOLOR-p6, YOLOXm and YOLOv7

5. Future Work

Further improvements are expected to result in an improved accuracy and faster frames per second rate while detecting players of both teams distinctly. This method could be refined more by also considering to track players and their movements which can be a great help in detecting off-sides too on real time, as offside detection is still one of the most challenging decisions in soccer at the moment, a goal ruled out wrongly because of error in offside detection by referee can change the game result. Furthermore, an application can be built on top of this model to distinguish the substitute players from the spectators so their movements can also be tracked closely. Semantics and instance segmentation can be used for that. The performance analysis of the models in this paper are being compared after training them on MS COCO dataset. These models can be trained on the custom data and then the FPS comparison between them can be observed. These custom trained model can be implemented on the embedded devices like Jetson Nano for various uses. The change in the FPS in those devices and the use of quantization techniques to deal with the FPS drop is yet another discussion. Additionally, a user friendly dashboard to interpret the information about game trend, insight and likelihood of goal by teams can be created.

6. Conclusion

Over the past few years, YOLO family has seen some tremendous research work as it has been proved to be a great resource for real-time object detection. Due to its fast & accurate detection, YOLO algorithm has huge potential for being used in the commercial applications. In this paper, different variants of the YOLO algorithms are analyzed for the accuracy and speed to yield the best performing model. YOLOv5, YOLO-X, YOLO-R and YOLOv7 models were applied to a soccer game play video to track the moving ball and the players in the playground. All the models performed well however YOLOv7 proved to be state of the art in terms of the accuracy showing *58% AP* whereas YOLOR-p6 which comes right after the former model and shows *56% AP*. On the other hand, YOLOv7 performs very poorly in terms of FPS demonstrating only 17 frames per second coming last against YOLO-R, YOLO-X and YOLOv5. Our work proves that where YOLOv7 is not the fastest on our data however due to its accuracy, it can be used in commercial applications like User friendly Soccer analyzer dashboard for insight analysis and likelihood of goal. In edge devices, there is a limitation of using YOLOv7 as we need at least 5 FPS for certain applications however the small variant YOLOv7.pt can only provide 3 to 4 FPS that are not enough. After some modifications, it can also be used in embedded devices to support applications like retail store monitoring and autonomous robots & vehicles.

REFERENCES

[1] A. B. H.-Y. M. L. Chien-Yao Wang, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors".

[2] S. D. R. G. A. F. Joseph Redmon, "You Only Look Once: Unified, Real-Time Object Detection".

[3] G. C. J. W. X. Y. J. H. Xingxing Xie, "Oriented R-CNN for Object Detection".

[4] A. F. Joseph Redmon, "YOLO9000: Better, Faster, Stronger".

[5] H. L. K. L. L. C. Y. L. Renjie Xu, "A Forest Fire Detection System Based on Ensemble Learning".

[6] S. Z. C. Z. R. L. Xiqi Wang, "R-YOLO: A Real-Time Text Detector for Natural Scenes with Arbitrary Rotation".

[7] I.-H. Y. a. H.-Y. M. L. Chien-Yao Wang, "You Only Learn One Representation: Unified Network for Multiple Tasks".

[8] S. L. F. W. Z. L. J. S. Zheng Ge, "YOLOX: Exceeding YOLO Series in 2021".

[9] J.-w. H. S. L. Y. B. J. P. Youngwan Lee, An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection.

[10] A. B. H.-Y. M. L. Chien-Yao Wang, Scaled-YOLOv4: Scaling Cross Stage Partial Network.

[11] V. V. S. I. J. S. a. Z. W. Christian Szegedy, Rethinking the Inception Architecture for Computer Vision.

[12] Y. L. G. P. Z. L. J. E. H. K. Q. W. Gao Huang, Snapshot Ensembles: Train 1, get M for free.

[13] A. T. a. H. Valpola, Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results.

[14] P. I. D. P. D. P. V. a. A. G. W. Timur Garipov, Loss surfaces, mode connectivity, and fast ensembling of DNNs.

[15] D. P. T. G. D. V. a. A. G. W. Pavel Izmailov, Averaging weights leads to wider optima and better generalization.

[16] X. Z. Y. Z. J. H. G. D. a. J. S. Xiaohan Ding, Scaling up your kernels to 31x31: Revisiting large kernel design in CNNs.

[17] S. X. P. G. Z. Z. a. Z. T. Chen-Yu Lee, Deeply-supervised nets.

[18] X. Z. N. M. J. H. G. D. J. S. Xiaohan Ding, RepVGG: Making VGG-style ConvNets Great Again.

AUTHORS

Ismat Saira Gillani is a graduate student of AI and Robotic Engineering at Columbus State University, USA. She is passionate about Computer Vision, Robotics and Machine Learning. She is always ready to adapt new situation, solve problems efficiently and achieve productivity goals. She focuses on making use of SoTA algorithms to build applications and design customized AI pipelines to provide the best possible AI solutions and generate measurable business value.

Muhammad Rizwan Munawar has received his BS in Computer Science from the COMSATS University Islamabad, Pakistan, currently he is a computer vision Engineer in a Multi-National company. He is obsessed with Machine Vision and Embedded Systems, focusing on the development of vision base applications, refining model architectures for research tasks, and developing custom modules that are suitable for his research.

Muhammad Talha is a Electrical and Telecom Engineer from Pakistan. He is passionate about Computer Vision, Machine Learning, Deep Learning, Point Cloud Processing and Video Compression. Currently, he is pursuing PhD from University of Missouri Kansas City, United States.

Muhammad Salman Azhar, Double Gold Medalist is Computer Science graduate from COMSATS Pakistan, is currently doing his Master's in AI from Air University Islamabad, Pakistan. He is an expert advisor of Financial Markets. Love building Trading Bots. He is a skilled Developer and passionate about AI, VR and Robotics.

Yousra Mashkoor is a Software Engineer, She's an active OpenSource contributor and is one of the top 110 Github contributors from Pakistan. Yousra is an ardent individual who's passionate about emerging high tech and how it is revolutionizing. She has expertise in Blockchain, Cloud Architecture, and Data Science.

Muhammad Sami uddin is highly passionate in Computer Vision and Deep Learning. He has built numerous Computer vision-based projects. Enthusiastic learner with mathematical background. Currently working on 3D computer vision models.

Usama Zafar is a Software engineer from University of Faisalabad, Pakistan. He loves to code and solve problems. He is passionate about Computer Vision, Machine learning and Artificial Intelligence. He is obsessed with Computer vision and Deep Learning.