YOLOX: Exceeding YOLO Series in 2021

Zheng Ge*  Songtao Liu†  Feng Wang  Zeming Li  Jian Sun
Megvii Technology
{gezheng, liusongtao, wangfeng02, lizeming, sunjian}@megvii.com

Figure 1: Speed-accuracy trade-off of accurate models (top) and Size-accuracy curve of lite models on mobile devices (bottom) for YOLOX and other state-of-the-art object detectors.

Abstract

In this report, we present some experienced improvements to YOLO series, forming a new high-performance detector—YOLOX. We switch the YOLO detector to an anchor-free manner and conduct other advanced detection techniques, i.e., a decoupled head and the leading label assignment strategy SimOTA to achieve state-of-the-art results across a large scale range of models: For YOLO-Nano with only 0.91M parameters and 1.08G FLOPs, we get 25.3% AP on COCO, surpassing NanoDet by 1.8% AP; for YOLOv3, one of the most widely used detectors in industry, we boost it to 47.3% AP on COCO, outperforming the current best practice by 3.0% AP; for YOLOX-L with roughly the same amount of parameters as YOLOv4-CSP, YOLOv5-L, we achieve 50.0% AP on COCO at 13.7 ms.

1. Introduction

With the development of object detection, YOLO series [23, 24, 25, 1, 7] always pursue the optimal speed and accuracy trade-off for real-time applications. They extract the most advanced detection technologies available at the time (e.g., anchors [26] for YOLOv2 [24], Residual Net [9] for YOLOv3 [25]) and optimize the implementation for best practice. Currently, YOLOv5 [7] holds the best trade-off performance with 48.2% AP on COCO at 13.7 ms. Nevertheless, over the past two years, the major advances in object detection academia have focused on anchor-free detectors [29, 40, 14], advanced label assignment strategies [37, 36, 12, 41, 22, 4], and end-to-end (NMS-free) detectors [2, 32, 39]. These have not been integrated into YOLO families yet, as YOLOv4 and YOLOv5 researchers in practical scenes, and we also provide deploy versions with ONNX, TensorRT, NCNN, and Openvino supported. Source code is at https://github.com/Megvii-BaseDetection/YOLOX.
are still anchor-based detectors with hand-crafted assigning rules for training.

That’s what brings us here, delivering those recent advancements to YOLO series with experienced optimization. Considering YOLOv4 and YOLOv5 may be a little over-optimized for the anchor-based pipeline, we choose YOLOv3 [25] as our start point (we set YOLOv3-SPP as the default YOLOv3). Indeed, YOLOv3 is still one of the most widely used detectors in the industry due to the limited computation resources and the insufficient software support in various practical applications.

As shown in Fig. 1, with the experienced updates of the above techniques, we boost the YOLOv3 to 47.3% AP (YOLOX-DarkNet53) on COCO with 640×640 resolution, surpassing the current best practice of YOLOv3 (44.3% AP, ultralytics version⁵) by a large margin. Moreover, when switching to the advanced YOLOv5 architecture that adopts an advanced CSPNet [31] backbone and an additional PAN [19] head, YOLOX-L achieves 50.0% AP on COCO with 640×640 resolution, outperforming the counterpart YOLOv5-L by 1.8% AP. We also test our design strategies on models of small size. YOLOX-Tiny and YOLOX-Nano (only 0.91M Parameters and 1.08G FLOPs) outperform the corresponding counterparts YOLOv4-Tiny and NanoDet³ by 10% AP and 1.8% AP, respectively.

We have released our code at https://github.com/Megvii-BaseDetection/YOLOX, with ONNX, TensorRT, NCNN and Openvino supported. One more thing worth mentioning, we won the 1st Place on Streaming Perception Challenge (Workshop on Autonomous Driving at CVPR 2021) using a single YOLOX-L model.

2. YOLOX

2.1. YOLOX-DarkNet53

We choose YOLOv3 [25] with Darknet53 as our baseline. In the following part, we will walk through the whole system designs in YOLOX step by step.

Implementation details Our training settings are mostly consistent from the baseline to our final model. We train the models for a total of 300 epochs with 5 epochs warm-up on COCO train2017 [17]. We use stochastic gradient descent (SGD) for training. We use a learning rate of \( lr \times \text{BatchSize}/64 \) (linear scaling [8]), with a initial \( lr = 0.01 \) and the cosine lr schedule. The weight decay is 0.0005 and the SGD momentum is 0.9. The batch size is 128 by default to typical 8-GPU devices. Other batch sizes include single GPU training also work well. The input size is evenly drawn from 448 to 832 with 32 strides. FPS and

| Models                      | Coupled Head | Decoupled Head |
|-----------------------------|--------------|----------------|
| Vanilla YOLO                | 38.5         | 39.6           |
| End-to-end YOLO             | 34.3 (-4.2)  | 38.8 (-0.8)    |

Table 1: The effect of decoupled head for end-to-end YOLO in terms of AP (%) on COCO.

latency in this report are all measured with FP16-precision and batch=1 on a single Tesla V100.

YOLOv3 baseline Our baseline adopts the architecture of DarkNet53 backbone and an SPP layer, referred to YOLOv3-SPP in some papers [1, 7]. We slightly change some training strategies compared to the original implementation [25], adding EMA weights updating, cosine lr schedule, IoU loss and IoU-aware branch. We use BCE Loss for training \( cls \) and \( obj \) branch, and IoU Loss for training \( reg \) branch. These general training tricks are orthogonal to the key improvement of YOLOX, we thus put them on the baseline. Moreover, we only conduct RandomHorizontalFlip, ColorJitter and multi-scale for data augmentation and discard the RandomResizedCrop strategy, because we found the RandomResizedCrop is kind of overlapped with the planned mosaic augmentation. With those enhancements, our baseline achieves 38.5% AP on COCO \( val \), as shown in Tab. 2.

Decoupled head In object detection, the conflict between classification and regression tasks is a well-known problem [27, 34]. Thus the decoupled head for classification and localization is widely used in the most of one-stage and two-stage detectors [16, 29, 35, 34]. However, as YOLO's and YOLO series' backbones and feature pyramids (e.g., FPN [13], PAN [20]) continuously evolving, their detection heads remain coupled as shown in Fig. 2.

Our two analytical experiments indicate that the coupled detection head may harm the performance. 1). Replacing YOLO's head with a decoupled one greatly improves the converging speed as shown in Fig. 3. 2). The decoupled head is essential to the end-to-end version of YOLO (will be described next). One can tell from Tab. 1, the end-to-end property decreases by 4.2% AP with the coupled head, while the decreasing reduces to 0.8% AP for a decoupled head. We thus replace the YOLO detect head with a lite decoupled head as in Fig. 2. Concretely, it contains a \( 1 \times 1 \) conv layer to reduce the channel dimension, followed by two parallel branches with two \( 3 \times 3 \) conv layers respectively. We report the inference time with batch=1 on V100 in Tab. 2 and the lite decoupled head brings additional 1.1 ms (11.6 ms v.s. 10.5 ms).

---

²https://github.com/ultralytics/yolov3
³https://github.com/RangiLyu/nanodet
Figure 2: Illustration of the difference between YOLOv3 head and the proposed decoupled head. For each level of FPN feature, we first adopt a $1 \times 1$ conv layer to reduce the feature channel to 256 and then add two parallel branches with two $3 \times 3$ conv layers each for classification and regression tasks respectively. IoU branch is added on the regression branch.

Figure 3: Training curves for detectors with YOLOv3 head or decoupled head. We evaluate the AP on COCO val every 10 epochs. It is obvious that the decoupled head converges much faster than the YOLOv3 head and achieves better result finally.

**Strong data augmentation** We add Mosaic and MixUp into our augmentation strategies to boost YOLOX’s performance. Mosaic is an efficient augmentation strategy proposed by ultralytics-YOLOv3\(^2\). It is then widely used in YOLOv4 [1], YOLOv5 [7] and other detectors [3]. MixUp [10] is originally designed for image classification task but then modified in BoF [38] for object detection training. We adopt the MixUp and Mosaic implementation in our model and close it for the last 15 epochs, achieving 42.0% AP in Tab. 2. After using strong data augmentation, we found ImageNet pre-training is no more beneficial, we thus train all the following models from scratch.

**Anchor-free** Both YOLOv4 [1] and YOLOv5 [7] follow the original anchor-based pipeline of YOLOv3 [25]. However, the anchor mechanism has many known problems. First, to achieve optimal detection performance, one needs to conduct clustering analysis to determine a set of optimal anchors before training. Those clustered anchors are domain-specific and less generalized. Second, anchor mechanism increases the complexity of detection heads, as well as the number of predictions for each image. On some edge AI systems, moving such large amount of predictions between devices (e.g., from NPU to CPU) may become a potential bottleneck in terms of the overall latency.

Anchor-free detectors [29, 40, 14] have developed rapidly in the past two year. These works have shown that the performance of anchor-free detectors can be on par with anchor-based detectors. Anchor-free mechanism significantly reduces the number of design parameters which need heuristic tuning and many tricks involved (e.g., Anchor Clustering [24], Grid Sensitive [11].) for good performance, making the detector, especially its training and decoding phase, considerably simpler [29].

Switching YOLO to an anchor-free manner is quite simple. We reduce the predictions for each location from 3 to 1 and make them directly predict four values, i.e., two offsets in terms of the left-top corner of the grid, and the height and width of the predicted box. We assign the center lo-
categorization of each object as the positive sample and pre-define a scale range, as done in [29], to designate the FPN level for each object. Such modification reduces the parameters and GFLOPs of the detector and makes it faster, but obtains better performance – 42.9% AP as shown in Tab. 2.

**Multi positives** To be consistent with the assigning rule of YOLOv3, the above anchor-free version selects only ONE positive sample (the center location) for each object meanwhile ignores other high quality predictions. However, optimizing those high quality predictions may also bring beneficial gradients, which may alleviates the extreme imbalance of positive/negative sampling during training. We simply assigns the center 3×3 area as positives, also named “center sampling” in FCOS [29]. The performance of the detector improves to 45.0% AP as in Tab. 2, already surpassing the current best practice of ultralytics-YOLOv3 (44.3% AP).

**SimOTA** Advanced label assignment is another important progress of object detection in recent years. Based on our own study OTA [4], we conclude four key insights for an advanced label assignment: 1). loss/quality aware, 2). center prior, 3). dynamic number of positive anchors† for each ground-truth (abbreviated as dynamic top-k), 4). global view. OTA meets all four rules above, hence we choose it as a candidate label assigning strategy.

Specifically, OTA [4] analyzes the label assignment from a global perspective and formulate the assigning procedure as an Optimal Transport (OT) problem, producing the SOTA performance among the current assigning strategies [12, 41, 36, 22, 37]. However, in practice we found solving OT problem via Sinkhorn-Knopp algorithm brings 25% extra training time, which is quite expensive for training 300 epochs. We thus simplify it to dynamic top-k strategy, named SimOTA, to get an approximate solution.

We briefly introduce SimOTA here. SimOTA first calculates pair-wise matching degree, represented by cost [4, 5, 12, 2] or quality [33] for each prediction-gt pair. For example, in SimOTA, the cost between gt $g_i$ and prediction $p_j$ is calculated as:

$$c_{ij} = L_{ij}^{cls} + \lambda L_{ij}^{reg},$$

where $\lambda$ is a balancing coefficient. $L_{ij}^{cls}$ and $L_{ij}^{reg}$ are classification loss and regression loss between gt $g_i$ and prediction $p_j$. Then, for gt $g_i$, we select the top $k$ predictions with the least cost within a fixed center region as its positive samples. Finally, the corresponding grids of those positive predictions are assigned as positives, while the rest grids are negatives. Noted that the value $k$ varies for different ground-truth. Please refer to Dynamic $k$ Estimation strategy in OTA [4] for more details.

SimOTA not only reduces the training time but also avoids additional solver hyperparameters in Sinkhorn-Knopp algorithm. As shown in Tab. 2, SimOTA raises the detector from 45.0% AP to 47.3% AP, higher than the SOTA ultralytics-YOLOv3 by 3.0% AP, showing the power of the advanced assigning strategy.

**End-to-end YOLO** We follow [39] to add two additional conv layers, one-to-one label assignment, and stop gradient. These enable the detector to perform an end-to-end manner, but slightly decreasing the performance and the inference speed, as listed in Tab. 2. We thus leave it as an optional module which is not involved in our final models.

### 2.2. Other Backbones

Besides DarkNet53, we also test YOLOX on other backbones with different sizes, where YOLOX achieves consistent improvements against all the corresponding counterparts.

---

†The term “anchor” refers to “anchor point” in the context of anchor-free detectors and “grid” in the context of YOLO.
Table 3: Comparison of YOLOX and YOLOv5 in terms of AP (%) on COCO. All the models are tested at 640 × 640 resolution, with FP16-precision and batch=1 on a Tesla V100.

| Models   | AP (%) | Parameters | GFLOPs | Latency |
|----------|--------|------------|--------|---------|
| YOLOv5-S | 36.7   | 7.3 M      | 17.1   | 8.7 ms  |
| YOLOX-S  | 39.6 (+2.9) | 9.0 M | 26.8 | 9.8 ms  |
| YOLOv5-M | 44.5   | 21.4 M     | 51.4   | 11.1 ms |
| YOLOX-M  | 46.4 (+1.9) | 25.3 M | 73.8 | 12.3 ms |
| YOLOv5-L | 48.2   | 47.1 M     | 115.6  | 13.7 ms |
| YOLOX-L  | 50.0 (+1.8) | 54.2 M | 155.6 | 14.5 ms |
| YOLOv5-X | 50.4   | 87.8 M     | 219.0  | 16.0 ms |
| YOLOX-X  | 51.2 (+0.8) | 99.1 M | 281.9 | 17.3 ms |

Table 4: Comparison of YOLOX-Tiny and YOLOX-Nano and the counterparts in terms of AP (%) on COCO val. All the models are tested at 416 × 416 resolution.

| Models       | AP (%) | Parameters | GFLOPs |
|--------------|--------|------------|--------|
| YOLOv4-Tiny  | 21.7   | 6.06 M     | 6.96   |
| PPyolo-Tiny  | 22.7   | 4.20 M     |        |
| YOLOX-Tiny   | 32.8 (+10.1) | 5.06 M | 6.45   |
| NanoDet      | 23.5   | 0.95 M     | 1.20   |
| YOLOX-Nano   | 25.3 (+1.8) | 0.91 M | 1.08   |

Table 5: Effects of data augmentation under different model sizes. “Scale Jit.” stands for the range of scale jittering for mosaic image. Instance mask annotations from COCO trainval are used when adopting Copypaste.

| Models       | Scale Jit. | Extra Aug. | AP (%) |
|--------------|------------|------------|--------|
| YOLOX-Nano   | [0.5, 1.5] | MixUp      | 25.3   |
|              | [0.1, 2.0] |            | 24.0 (+1.3) |
| YOLOX-L      | [0.1, 2.0] | MixUp      | 48.6   |
|              | [0.1, 2.0] | Copypaste  [6] | 49.5 (+0.9) |

3. Comparison with the SOTA

There is a tradition to show the SOTA comparing table as in Tab. 6. However, keep in mind that the inference speed of the models in this table is often uncontrolled, as speed varies with software and hardware. We thus use the same hardware and code base for all the YOLO series in Fig. 1, plotting the somewhat controlled speed/accuracy curve.

We notice that there are some high performance YOLO series with larger model sizes like Scale-YOLOv4 [30] and YOLOv5-P6 [7]. And the current Transformer based detectors [21] push the accuracy-SOTA to ~60 AP. Due to the time and resource limitation, we did not explore those important features in this report. However, they are already in our scope.

4. 1st Place on Streaming Perception Challenge (WAD at CVPR 2021)

Streaming Perception Challenge on WAD 2021 is a joint evaluation of accuracy and latency through a recently proposed metric: streaming accuracy [15]. The key insight be-
Table 6: Comparison of the speed and accuracy of different object detectors on COCO 2017 test-dev. We select all the models trained on 300 epochs for fair comparison.

| Method                  | Backbone           | Size | FPS  |
|-------------------------|--------------------|------|------|
|YOLOv3 + ASFF* [18]     | Darknet-53         | 608  | 45.5 |
|YOLOv3 + ASFF* [18]     | Darknet-53         | 800  | 29.4 |
|EfficientDet-D0 [28]    | Efficient-B0       | 512  | 98.0 |
|EfficientDet-D1 [28]    | Efficient-B1       | 640  | 74.1 |
|EfficientDet-D2 [28]    | Efficient-B2       | 768  | 56.5 |
|EfficientDet-D3 [28]    | Efficient-B3       | 896  | 34.5 |
|PP-YOLOv2 [11]          | ResNet50-vd-dcn    | 640  | 68.9 |
|PP-YOLOv2 [11]          | ResNet101-vd-dcn   | 640  | 50.3 |
|YOLOv4 [1]              | CSPDarknet-53      | 608  | 62.0 |
|YOLOv4-CSP [30]         | Modified CSP       | 640  | 73.0 |
|YOLOv3-ultralytics²     | Darknet-53         | 640  | 95.2 |
|YOLOv5-M [7]            | Modified CSP v5    | 640  | 90.1 |
|YOLOv5-L [7]            | Modified CSP v5    | 640  | 73.0 |
|YOLOv5-X [7]            | Modified CSP v5    | 640  | 62.5 |
|YOLOX-DarkNet53         | Darknet-53         | 640  | 90.1 |
|YOLOX-M                 | Modified CSP v5    | 640  | 81.3 |
|YOLOX-L                 | Modified CSP v5    | 640  | 69.0 |
|YOLOX-X                 | Modified CSP v5    | 640  | 57.8 |

hind this metric is to jointly evaluate the output of the entire perception stack at every time instant, forcing the stack to consider the amount of streaming data that should be ignored while computation is occurring [15]. We found that the best trade-off point for the metric on 30 FPS data stream is a powerful model with the inference time \( \leq 33 \text{ms} \). So we adopt a YOLOX-L model with TensorRT to produce our final model for the challenge to win the 1st place. Please refer to the challenge website⁵ for more details.

5. Conclusion

In this report, we present some experienced updates to YOLO series, which forms a high-performance anchor-free detector called YOLOX. Equipped with some recent advanced detection techniques, i.e., decoupled head, anchor-free, and advanced label assigning strategy, YOLOX achieves a better trade-off between speed and accuracy than other counterparts across all model sizes. It is remarkable that we boost the architecture of YOLOv3, which is still one of the most widely used detectors in industry due to its broad compatibility, to 47.3% AP on COCO, surpassing the current best practice by 3.0% AP. We hope this report can help developers and researchers get better experience in practical scenes.

Acknowledgements

This research was supported by National Key R&D Program of China (No. 2017YFA0700800). It was also funded by China Postdoctoral Science Foundation (2021M690375) and Beijing Postdoctoral Research Foundation.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020. 1, 2, 3, 6
[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In ECCV, 2020. 1, 4
[3] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. In CVPR, 2021. 3
[4] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In CVPR, 2021. 1, 4
[5] Zheng Ge, Jianfeng Wang, Xin Huang, Songtao Liu, and Osamu Yoshie. Lla: Loss-aware label assignment for dense pedestrian detection. arXiv preprint arXiv:2101.04307, 2021. 4
[6] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple

⁵https://eval.ai/web/challenges/challenge-page/800/overview
copy-paste is a strong data augmentation method for instance segmentation. In CVPR, 2021. 5

[7] glenn jocher et al. yolov5. https://github.com/ultralytics/yolov5, 2021. 1, 2, 3, 5, 6

[8] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large mini-batch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677, 2017. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqiang Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, 2016. 1

[10] Zhang Hongyi, Cisse Moustapha, N. Dauphin Yann, and David Lopez-Paz. mixup: Beyond empirical risk minimization. ICLR, 2018. 3

[11] Xin Huang, Xinxin Wang, Wenyu Lv, Xiaying Bai, Xiang Long, Kaipeng Deng, Qingqing Dang, Shumin Han, Qiwen Liu, Xiaoguang Hu, et al. Pp-yolov2: A practical object detector. arXiv preprint arXiv:2104.10419, 2021. 3, 6

[12] Kang Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In ECCV, 2020. 1, 4

[13] Seung-Wook Kim, Hyong-Keun Kook, Jee-Young Sun, Mun-Cheon Kang, and Sung-Jea Ko. Parallel feature pyramid network for object detection. In ECCV, 2018. 2

[14] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In ECCV, 2018. 1, 3

[15] Mengtian Lai, Yuxiong Wang, and Deva Ramanan. Towards streaming perception. In ECCV, 2020. 5, 6

[16] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In NeurIPS, 2015. 1

[17] Guanglu Song, Yu Liu, and Xiaogang Wang. Revisiting the sibling head in object detector. In CVPR, 2020. 2

[18] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. arXiv preprint arXiv:2101.08036, 2020. 1, 5, 6

[19] Fcos: Fully convolutional one-stage object detection. In ICCV, 2019. 1, 2, 3, 4

[20] Jia Gao, Ming-xi Xu, and Yiran Chen. Class balancing for object detection. In CVPR workshops, 2020. 2, 5

[21] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018. 1, 2, 3

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In CVPR, 2016. 1

[23] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In CVPR, 2017. 1, 3

[24] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In CVPR, 2017. 1, 3