DAMO-YOLO : A Report on Real-Time Object Detection Design

Xianzhe Xu*, Yiqi Jiang*, Weihua Chen*, Yilun Huang*,Yuan Zhang*, Xiuyu Sun†
Alibaba Group

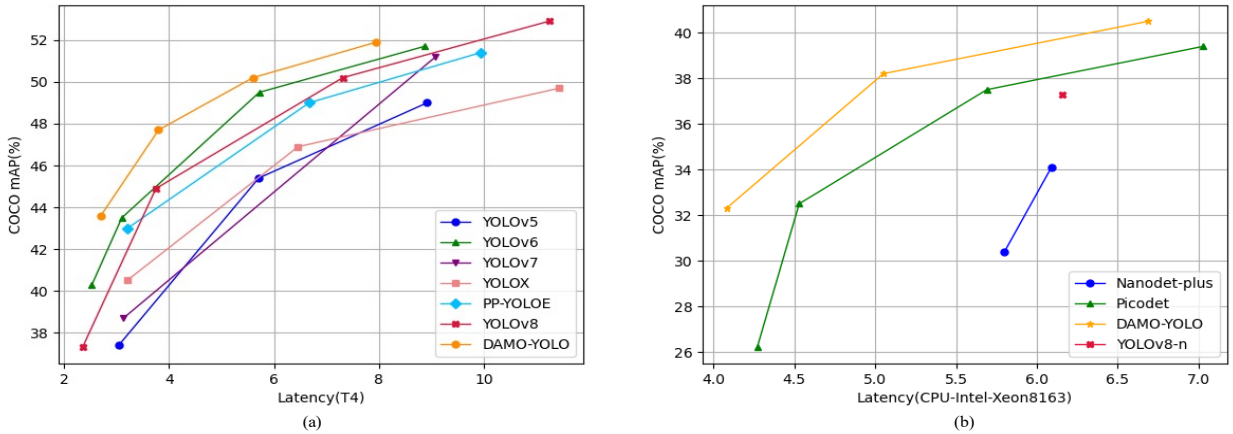

Figure 1. Latency-accuracy trade-off of models for DAMO-YOLO and other state-of-the-art object detectors. (a) Comparision between the DAMO-YOLO-T/S/M/L and other SOTA general detectors on T4-GPU. (b) Comparision between the DAMO-YOLO-Ns/Nm/Nl and other SOTA light weight detectors on x86-CPU.

Abstract

*In this report, we present a fast and accurate object detection method dubbed **DAMO-YOLO**, which achieves higher performance than the state-of-the-art YOLO series. DAMO-YOLO is extended from YOLO with some new technologies, including Neural Architecture Search (NAS), efficient Reparameterized Generalized-FPN (RepGFPN), a lightweight head with AlignedOTA label assignment, and distillation enhancement. In particular, we use MAE-NAS, a method guided by the principle of maximum entropy, to search our detection backbone under the constraints of low latency and high performance, producing ResNet-like / CSP-like structures with spatial pyramid pooling and focus modules. In the design of necks and heads, we follow the rule of "large neck, small head". We import Generalized-FPN with accelerated queen-fusion to build the detector neck and upgrade its CSPNet with efficient layer aggregation networks (ELAN) and reparameterization. Then we investigate how detector head size affects detection performance and find that a heavy neck with only one task projection layer would yield better results. In addition, AlignedOTA is proposed to solve the misalignment problem in label assignment. And a distillation schema is introduced to improve performance to a higher level. Based on these new techs, we build a suite of models at various scales to meet the needs of different scenarios. For general industry requirements, we propose DAMO-YOLO-T/S/M/L. They can achieve 43.6/47.7/50.2/51.9 mAPs on COCO with the latency of 2.78/3.83/5.62/7.95 ms on T4 GPUs respectively. Additionally, for edge devices with limited computing power, we have also proposed DAMO-YOLO-Ns/Nm/Nl lightweight models. They can achieve 32.3/38.2/40.5 mAPs on COCO with the latency of 4.08/5.05/6.69 ms on X86-CPU. Our proposed general and lightweight models have outperformed other YOLO series models in their respective application scenarios. The code is available at https://github.com/tinyvision/damo-yolo.*

*These authors contributed equally to this work
†Corresponding author xiuyu.sxy@alibaba-inc.com

Table 1. CSP-Darknet vs MAE-NAS Backbone under DAMO-YOLO framework with different scales.

Scale	Depth	Backbone	AP	Latency(ms)
S	25	CSP-Darknet	44.9	3.92
S	25	MAE-Res	45.6	3.83
S	25	MAE-CSP	45.3	3.79
M	35	MAE-Res	48.0	5.64
M	35	MAE-CSP	48.7	5.60

Table 2. CSP-Darknet vs MAE-NAS Backbone under DAMO-YOLO framework with different scales. * denotes latency evaluated on X86-CPU.

Scale	Depth	Backbone	AP	Latency(ms)
N	18	MAE-Mob	38.2	5.05*
N	18	MAE-Res	37.4	5.89*
S	25	CSP-Darknet	44.9	3.92
S	25	MAE-Res	45.6	3.83
S	25	MAE-CSP	45.3	3.79
M	35	MAE-Res	48.0	5.64
M	35	MAE-CSP	48.7	5.60

1. Introduction

Recently, researchers have developed object detection methods in huge progress [1, 9, 11, 23, 24, 27, 33]. While the industry pursues high-performance object detection methods with real-time constraints, researchers focus on designing one-stage detectors [1, 22–24, 26] with efficient network architectures [4, 16, 17, 29, 30] and advanced training stages [10, 17, 21, 26, 31]. Especially, YOLOv5/6/7 [18, 32, 34], YOLOX [9] and PP-YOLOE [38] have achieved significant AP-Latency trade-offs on COCO, making YOLO series object detection methods widely used in the industry.

Although object detection has achieved great progress, there are still new techs that can be brought in to further improve performance. Firstly, the network structure plays a critical role in object detection. Darknet holds a dominant position in the early stages of YOLO history [1, 9, 24–26, 32]. Recently, some works have investigated other efficient networks for their detectors, *i.e.*, YOLOv6 [18] and YOLOv7 [34]. However, these networks are still manually designed. Thanks to the development of the Neural Architecture Search (NAS), there are many detection-friendly network structures found through the NAS techs [4, 16, 30], which have shown great superiority over previous manually designed networks. Therefore, we take advantage of the NAS techs and import MAE-NAS [30][1] for our DAMO-YOLO. MAE-NAS is a heuristic and training-free neural architecture search method without supernet dependence and can be utilized to archive backbones at different scales. It can produce ResNet-like / CSP-like structures with spatial pyramid pooling and focus modules.

Secondly, it is crucial for a detector to learn sufficient fused information between high-level semantic and low-level spatial features, which makes the detector neck to be a vital part of the whole framework. The importance of neck has also been discussed in other works [10, 17, 31, 36]. Feature Pyramid Network (FPN) [10] has been proved effective to fuse multi-scale features. Generalized-FPN (GFPN) [17] improves FPN with a novel queen-fusion. In DAMO-YOLO, we design a Reparameterized Generalized-FPN (RepGFPN). It is based on GFPN but involved in an accelerated queen-fusion, the efficient layer aggregation networks (ELAN) and re-parameterization.

To strike the balance between latency and performance, we conducted a series of experiments to verify the importance of the neck and head of the detector and found that "large neck, small head" would lead to better performance. Hence, we discard the detector head in previous YOLO series works [1, 9, 24–26, 32, 38], but only left a task projection layer. The saved calculations are moved to the neck part. Besides the task projection module, there is no other training layer in the head, so we named our detector head as ZeroHead. Coupled with our RepGFPN, ZeroHead achieves state-of-the-art performance, which we believe would bring some insights to other researchers.

In addition, the dynamic label assignment, such as OTA [8] and TOOD [7], is widely acclaimed and achieves significant improvement compared to the static label assignment [43]. However, the misalignment problem is still unsolved in these works. We propose a better solution called AlignOTA to balance the importance of classification and regression, which can partly solve the problem.

At last, Knowledge Distillation (KD) has been proved effective in boosting small models by the larger model supervision. This tech does exactly fit the design of real-time object detection. Nevertheless, applying KD on YOLO series sometimes can not achieve significant improvements as hyperparameters are hard to optimize and features carry too much noise. In our DAMO-YOLO, we first make distillation great again on models of all sizes, especially on small ones.

As shown in Fig.1, with the above improvements, we proposed a series of general and lightweight models that exceed the state of the arts by a large margin.

In summary, the contributions are three-fold:

1. This paper proposes a new detector called **DAMO-YOLO**, which extends from YOLO but with more new

[1] https://github.com/alibaba/lightweight-neural-architecture-search. A demo can be found at the ModelScope.

techs, including MAE-NAS backbones, RepGFPN neck, ZeroHead, AlignedOTA and distillation enhancement.

2. DAMO-YOLO outperforms the state-of-the-art detectors (*e.g.* YOLO series) on public COCO datasets in both general and lightweight categories.

3. A suite of models with various scales is presented in DAMO-YOLO (tiny/small/medium) to support different deployments. The code and pre-trained models are released at https://github.com/tinyvision/damo-yolo, with ONNX and TensorRT supported.

2. DAMO-YOLO

In this section, we introduce each module of DAMO-YOLO in detail, including Neural Architecture Search (NAS) backbones, efficient Reparameterized Generalized-FPN (RepGFPN) neck, ZeroHead, AlignedOTA label assignment and distillation enhancement. The whole framework of DAMO-YOLO is displayed in Fig.3.

2.1. MAE-NAS Backbone

Previously, in real-time scenarios, designers relied on the Flops-mAP curve as a simple means of assessing model performance. However, the relationship between a model's flops and latency is not necessarily consistent. In order to enhance a model's real-world performance in industrial deployment, DAMO-YOLO prioritized the latency-MAP curve in the design process.

Based on this design principle, we use MAE-NAS [30] to obtain optimal networks under different latency budgets. MAE-NAS constructs an alternative proxy based on information theory to rank initialized networks without training. Therefore, the search process only takes a few hours, which is much lower than the training costs. Several basic search blocks are provided by MAE-NAS, such as Mob-block, Res-block and CSP-block, as shown in Fig.2. The Mob-block is a variant of MobileNetV3 [14] block, the Res-block is derived from ResNet [12], and the CSP-block is derived from CSPNet [35]. The full supported block list can be found in MAE-NAS repository[2].

We found that applying different kinds of blocks in models of different scales achieves better trade-offs for real-time inference. The performance comparisons among CSP-Darknet and our MAE-NAS backbones under our DAMO-YOLO with different scales are listed in Table.1. In this table, "MAE-Res" means we apply Res-block in the MAE-NAS backbones, and "MAE-CSP" means we apply CSP-block in it. Besides, "S" (Small) and "M" (Medium) represent different scales of backbones. As shown in Table.1, the MAE-CSP obtained by MAE-NAS technology outperforms manually designed CSP-Darknet in terms of both speed and accuracy, demonstrating the superiority of MAE-NAS technolog. Moreover, we can observe from Table.1 that using Res-block on smaller models can achieve a better trade-off between performance and speed than CSP-block, while using CSP-block can significantly outperform Res-block on larger and deeper networks. As a consequences, we use "MAE-Res" in "T" (Tiny) and "S" models and "MAE-CSP" in "M" and "L" models in the final setting.

When dealing with scenarios that only have limited computing power or GPU is not available, it is crucial to have models that meet strict requirements for computation and speed. To address this issue, we have designed a series of light weight model using Mob-Block. Mob-block is derived from MobleNetV3 [14], which can significantly decrease model computation and is friendly with CPU devices.

[2]https://github.com/alibaba/lightweight-neural-architecture-search/blob/main/tinynas/models/README.md#supported-blocks

Table 3. Ablation Study on the depth and width of our neck. "Depth" denotes the repeat times on the bottleneck of fusion block. "Width" indicates the channel dimensions of feature maps.

Depth	Width	Latency	FLOPs	AP
2	(192, 192, 192)	3.53	34.9	44.2
2	(128, 256, 512)	3.72	36.1	45.1
3	(160, 160, 160)	3.91	38.2	44.9
3	**(96, 192, 384)**	**3.83**	**37.8**	**45.6**
4	(64, 128, 256)	3.85	37.2	45.3

2.2. Efficient RepGFPN

Feature pyramid network aims to aggregate different resolution features extracted from the backbone, which has been proven to be a critical and effective part of object detection [10, 31, 36]. The conventional FPN [10] introduces a top-down pathway to fuse multi-scale features. Considering the limitation of one-way information flow, PAFPN [36] adds an additional bottom-up path aggregation network, but with higher computational costs. BiFPN [31] removes nodes that only have one input edge, and adds skip-link from the original input on the same level. In [17], Generalized-FPN (GFPN) is proposed to serve as neck and achieves SOTA performance, as it can sufficiently exchange high-level semantic information and low-level spatial information. In GFPN, multi-scale features are fused in both level features in previous and current layers. What's more, the $log_2(n)$ skip-layer connections provide more effective information transmission that can scale into deeper networks. When we directly replace modified-PANet with GFPN on modern YOLO-series models, we achieved higher precision. However, the latency of GFPN-

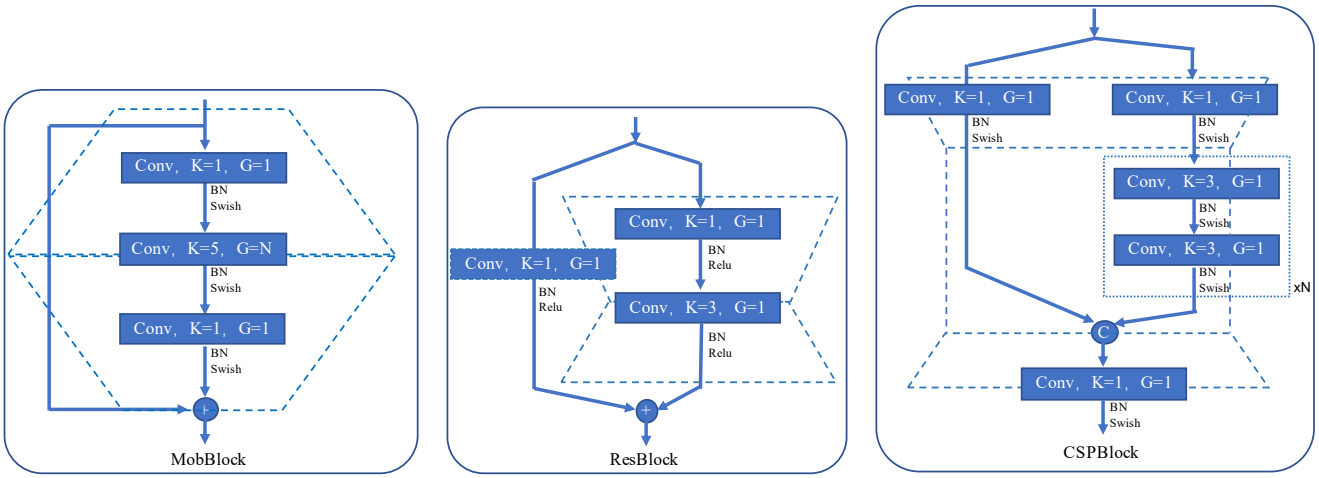

Figure 2. Different building block for MAE-NAS. (a) MobBlock is a variant of MobileNetV3 block and serves as the basic block for the lightweight model in DAMO-YOLO. (b) ResBlock is derived from ResNet. DAMO-YOLO-S and DAMO-YOLO-T is built upon it. (c) CSPBlock is derived from CSPNet, and due to its excellent performance in deep networks, it is used as a basic block in DAMO-YOLO-M and DAMO-YOLO-L models.

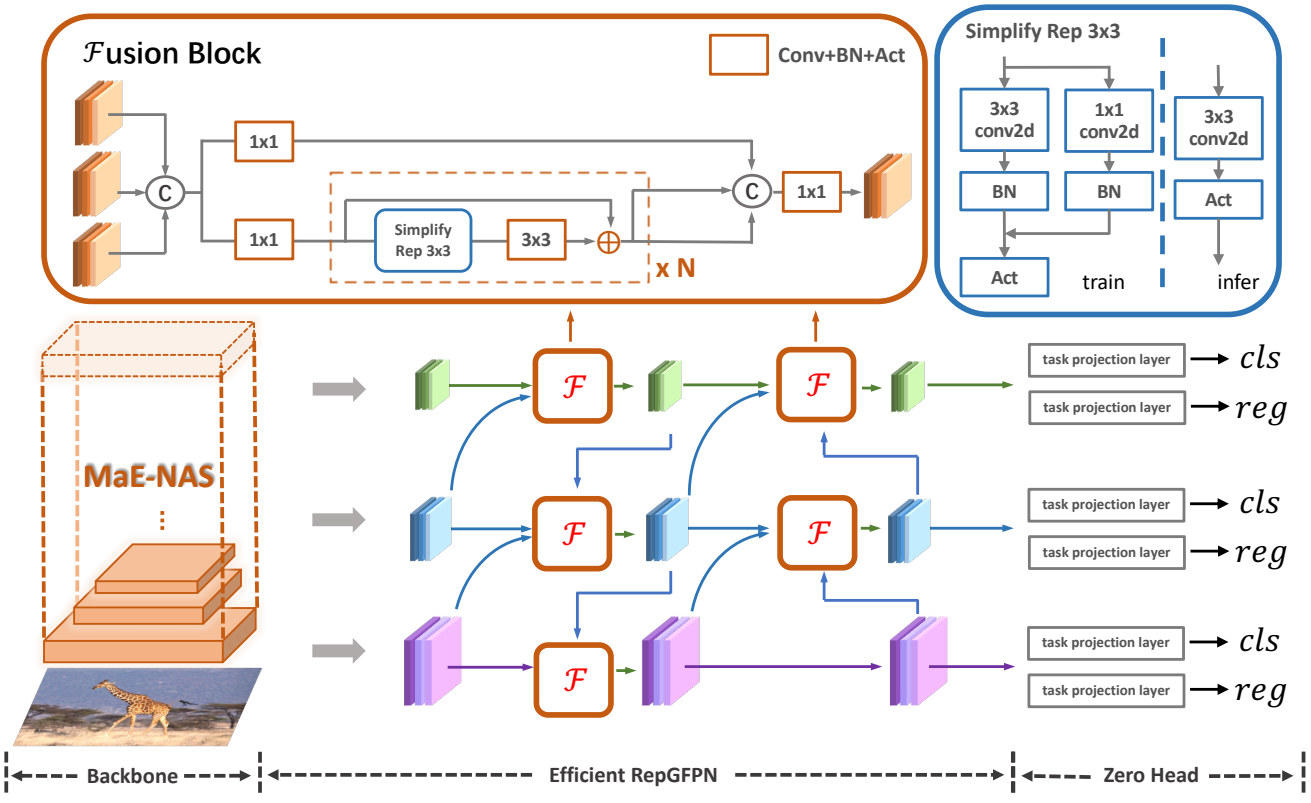

Figure 3. Overview of the network architecture of DAMO-YOLO. 1) MAE-NAS as backbone to extract multi-scale feature maps; 2) Efficient RepGFPN as neck to refine and fuse high-level semantic and low-level spatial features; 3) ZeroHead is presented which only contains a task projection layer for each loss.

Table 4. Ablation Study on the connection of queen-fusion. ↘ and ↗ denote the upsampling and downsampling operations respectively.

↘	↗	Latency	FLOPs	AP
		3.62	33.3	44.2
✓		4.19	37.7	44.5
	✓	**3.83**	**37.8**	**45.6**
✓	✓	4.58	42.8	45.9

Table 5. Ablation study on the feature fusion style. CSP denotes the Cross-Stage-Partial Connection. Reparam [5, 6] denotes applying re-parameter mechanism on the bottleneck of CSP. ELAN denotes the connections of efficient layer aggregation networks.

Merge−Style	Latency	FLOPs	AP
Conv	3.64	44.3	40.2
CSP	3.72	36.7	44.4
CSP + Reparam	3.72	36.7	45.0
CSP + Reparam + ELAN	**3.83**	**37.8**	**45.6**

based model is much higher than modified-PANet-based model. By analyzing the structure of GFPN, the reason can be attributed to the following aspects: 1) feature maps with different scales share the same dimension of channels; 2) the operation of queen-fusion can not meet the requirement for real-time detection model; 3) the convolution-based cross-scale feature fusion is not efficient.

Based on GFPN, we propose a novel Efficient-RepGFPN to meet the design of real-time object detection, which mainly consists of the following insights: 1) Due to the large difference in FLOPs from different scale feature maps, it is difficult to control the same dimension of channels shared by each scale feature map under the constraint of limited computation cost. Therefore, in the feature fusion of our neck, we adopt the setting of different scale feature maps with different dimensions of channels. Performance with the same and different channels as well as precision benefits from the Neck depth and width trade-offs are compared, Table.3 shows the results. We can see that by flexibly controlling the number of channels in different scales, we can achieve much higher accuracy than sharing the same channels at all scales. Best performance is obtained when depth equals 3 and width equals (96, 192, 384). 2) GFPN enhances feature interaction by queen-fusion, but it also brings lots of extra upsampling and downsampling operators. The benefits of those upsampling and downsampling operators are compared and results are shown in Table.4. We can see that the additional upsampling operator results in a latency increase of 0.6ms, while the accuracy improvement was only 0.3mAP, far less than the benefit of the additional downsampling operator. Therefore, under the constraints of real-time detection, we remove the extra upsampling operation in queen-fusion. 3) In the feature fusion block, we first replace original 3x3-convolution-based feature fusion with CSPNet and obtain 4.2 mAP gain. Afterward, we upgrade CSPNet by incorporating re-parameterization mechanism and connections of efficient layer aggregation networks (ELAN) [34]. Without bringing extra huge computation burden, we achieve much higher precision. The results of comparison are listed in Table.5.

Table 6. Studies on the balance between RepGFPN and ZeroHead.

Neck(width/depth)	Head(width/depth)	Latency(ms)	AP
(1.0/1.0)	**(1.0/0.0)**	**3.83**	**45.6**
(1.0/0.50)	(1.0/1.0)	3.79	44.9
(1.0/0.33)	(1.0/2.0)	3.85	43.7
(1.0/0.0)	(1.0/3.0)	3.87	41.2

2.3. ZeroHead and AlignOTA

In recent advancements of object detection, decoupled head is widely used [9, 18, 38]. With the decoupled head, those models achieve higher AP, while the latency grows significantly. To trade off the latency and the performance, we have conducted a series of experiments to balance the importance of neck and head, and the results are shown in Table.6. From the experiments, we find that "large neck, small head" would lead to better performance. Hence, we discard the decoupled head in previous works [9, 18, 38], but only left a task projection layer, *i.e.*, one linear layer for classification and one linear layer for regression. We named our head as ZeroHead as there is no other training layers in our head. ZeroHead can save computations for the heavy RepGFPN neck to the greatest extent. It is worth noticing that ZeroHead essentially can be considered as a coupled head, which is quite a difference from the decoupled heads in other works [9, 18, 32, 38]. In the loss after head, following GFocal [20], we use Quality Focal Loss (QFL) for classification supervision, and Distribution Focal Loss (DFL) and GIOU loss for regression supervision. QFL encourages to learn a joint representation of classification and localization quality. DFL provides more informative and precise bounding box estimations by modeling their locations as General distributions. The training loss of the proposed DAMO-YOLO is formulated as:

$$Loss = \alpha\, loss_{QFL} + \beta\, loss_{DFL} + \gamma\, loss_{GIOU} \quad (1)$$

Besides head and loss, label assignment is a crucial

Table 7. The comparison of different on MSCOCO val dataset.

Assigner	AP
ATSS [43]	43.1
simOTA [8]	44.2
TOOD [7]	45.4
AlignOTA	**45.6**

Table 8. Studies on the distillation methods for DAMO-YOLO on MSCOCO val dataset. The baseline of student is 38.2.

Methods	Epochs	AP
Mimicking [19]	36	40.2
MGD [39]	36	39.6
CWD [28]	36	**40.7**

component during detector training, which is responsible for assigning classification and regression targets to predefined anchors. Recently, dynamic label assignment such as OTA [8] and TOOD [7] is widely acclaimed and achieves significant improvements compares to static one [43]. Dynamic label assignment methods assign labels according to the assignment cost between prediction and ground truth, *e.g.*, OTA [8]. Although the alignment of classification and regression in loss is widely studied [7, 20], the alignment between classification and regression in label assignment is rarely mentioned in current works. The misalignment of classification and regression is a common issue in static assignment methods [43]. Though dynamic assignment alleviates the problem, it still exists due to the unbalance of classification and regression losses, *e.g.*, CrossEntropy and IoU Loss [40]. To solve this problem, we introduce the focal loss [22] into the classification cost, and use the IoU of prediction and ground truth box as the soft label, which is formulated as follows:

$$\begin{aligned} AssignCost &= C_{reg} + C_{cls} \\ \alpha &= IoU(reg_{gt}, reg_{pred}) \\ C_{reg} &= -ln(\alpha) \\ C_{cls} &= (\alpha - cls_{pred})^2 \times CE(cls_{pred}, \alpha) \end{aligned} \tag{2}$$

With this formulation, we are able to choose the classification and regression aligned samples for each target. Besides the aligned assignment cost, following OTA [8], we form the solution of aligned assignment cost from a global perspective. We name our label assignment as AlignOTA. The comparison of label assignment methods is conducted in Table.7. We can see that AlignOTA outperforms all other label assignment methods.

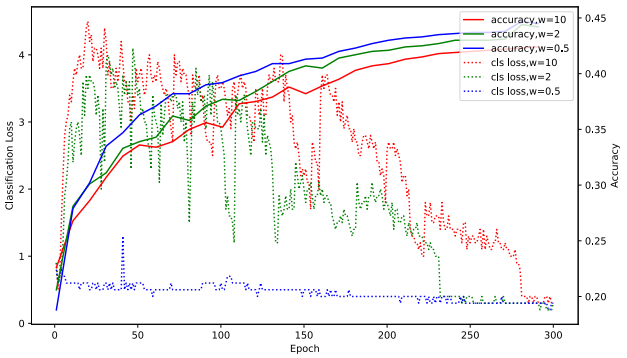

Figure 4. The classification loss and AP curves of distillation. The distillation loss weight is set to 0.5, 2, and 10 respectively. The classification loss has a significantly fast convergence with higher accuracy when the distillation loss weight is set to 0.5.

2.4. Distillation Enhancement

Knowledge Distillation (KD) [13] is an effective method to further boost the performance of pocket-size models. Nevertheless, applying KD on YOLO series sometimes can not achieve significant improvements as hyperparameters are hard to optimize and features carry too much noise. In DAMO-YOLO, we first make distillation great again on models of all sizes, especially on the small size. We adopt the feature-based distillation to transfer dark knowledge, which can distill both recognition and localization information in the intermediate feature maps [15]. We conduct fast validation experiments to choose a suitable distillation method for our DAMO-YOLO. The results are shown in Table.8. We conclude that CWD is more fit for our models, while MGD is worse than Mimicking as complex hyperparameters make it not general enough.

Our proposed distillation strategy is split into two stages: 1) Our teacher distills the student at the first stage (284 epochs) on **strong mosaic** domain. Facing the challenging augmented data distribution, the student can further extract information smoothly under the teacher's guidance. 2) The student finetunes itself on **no mosaic** domain at the second stage (16 epochs). The reason why we do not adopt distillation at this stage is that, in such a short period, the teacher's experience will damage the student's performance when he wants to pull the student in a strange domain (*i.e.*, no mosaic domain). A long-term distillation would weaken the damage but is expensive. So we choose a trade-off to make the student independent.

In DAMO-YOLO, the distillation is equipped with two advanced enhancements: 1) Align Module. On the one hand, it is a linear projection layer to adapt student feature's to the same resolution (C, H, W) as teacher's. On the other hand, forcing the student to approximate teacher feature directly leads to minor gains compared to the adaptive

Table 9. Comparison with the state-of-the-art single-model detectors on MSCOCO validation set. * denotes using distillation. Latency is tested by ourself on T4 GPUs using TensorRT engine in FP16, while other results are from the corresponding papers.

Method	Size	Latency(ms)	GFLOPs	Params(M)	FPS^{V100}	AP	AP^{50}	AP^{75}	AP^{S}	AP^{M}	AP^{L}
YOLOX-T	416	1.78	6.5	5.1	-	32.8	-	-	-	-	-
YOLOX-S	640	3.20	26.8	9.0	247	40.5	-	-	-	-	-
YOLOX-M	640	6.46	73.8	25.3	177	46.9	-	-	-	-	-
YOLOX-L	640	11.44	155.6	54.2	120	49.7	-	-	-	-	-
YOLOv5-N	640	2.23	4.5	1.9	-	28.0	45.7	-	-	-	-
YOLOv5-S	640	3.04	16.5	7.2	455	37.4	56.8	-	-	-	-
YOLOv5-M	640	5.71	49.0	21.2	263	45.4	64.1	-	-	-	-
YOLOv5-L	640	8.92	109.1	46.5	172	49.0	67.3	-	-	-	-
YOLOv6-T	640	2.53	36.7	15.0	-	40.3	56.6	-	-	-	-
YOLOv6-S	640	3.10	44.2	17.0	-	43.5	60.4	-	-	-	-
YOLOv6-M*	640	5.72	82.2	34.3	-	49.5	66.8	-	-	-	-
YOLOv6-L*	640	9.87	144.0	58.5	-	52.5	70.0	-	-	-	-
YOLOv7-T-silu	640	3.13	13.7	6.2	-	38.7	56.7	41.7	18.8	42.4	51.9
YOLOv7	640	9.08	104.7	36.9	-	51.2	69.7	55.9	31.8	55.5	65.0
YOLOE-S	640	3.21	17.4	7.9	333	43.0	60.5	46.6	23.2	46.4	56.9
YOLOE-M	640	6.67	49.9	23.4	208	49.0	66.5	53.0	28.6	52.9	63.8
YOLOE-L	640	9.94	110.1	52.2	149	51.4	68.9	55.6	31.4	55.3	66.1
DAMO-YOLO-T	640	2.78	18.1	8.5	397	42.0	58.0	45.2	23.0	46.1	58.5
DAMO-YOLO-T*	640	2.78	18.1	8.5	397	43.6	59.4	46.6	23.3	47.4	61.0
DAMO-YOLO-S	640	3.83	37.8	16.3	325	46.0	61.9	49.5	25.9	50.6	62.5
DAMO-YOLO-S*	640	3.83	37.8	16.3	325	47.7	63.5	51.1	26.9	51.7	64.9
DAMO-YOLO-M	640	5.62	61.8	28.2	233	49.2	65.5	53.0	29.7	53.1	66.1
DAMO-YOLO-M*	640	5.62	61.8	28.2	233	50.4	67.2	55.1	31.6	55.3	67.1
DAMO-YOLO-L	640	7.95	97.3	42.1	126	50.8	67.5	55.5	33.2	55.7	66.6
DAMO-YOLO-L*	640	7.95	97.3	42.1	126	51.9	68.5	56.7	33.3	57.0	67.6

Table 10. Comparison with the state-of-the-art light weight detectors on MSCOCO validation set. Latency is tested by ourself on Intel-8163 CPU with OpenVINO, while other results are from the corresponding papers.

Method	Size	Latency(ms)	GFLOPs	Params(M)	AP	AP^{50}	AP^{75}	AP^{S}	AP^{M}	AP^{L}
Picodet-xs	416	4.27	1.13	0.70	26.2	-	-	-	-	-
Picodet-s	416	4.53	1.65	1.18	32.5	-	-	-	-	-
Picodet-m	416	5.69	4.34	3.46	37.5	-	-	-	-	-
Picodet-l	416	7.03	7.10	5.80	39.4	-	-	-	-	-
Nanodet-plus-m	416	5.80	1.52	1.17	30.4	-	-	-	-	
Nanodet-plus-m1.5x	416	6.09	2.97	2.44	34.1	-	-	-	-	
yolov8n	640	6.16	8.7	3.2	37.3	–	-	-	-	-
DAMO-YOLO-Ns	416	4.08	1.56	1.41	32.3	47.7	34.2	12.3	34.8	51.8
DAMO-YOLO-Nm	416	5.05	3.69	2.14	38.2	54.7	40.8	20.2	41.7	57.6
DAMO-YOLO-Nl	416	6.69	6.04	5.69	40.5	57.6	43.3	20.7	44.5	60.9

imitation [37]. 2) Channel-wise Dynamic Temperature. Inspired by PKD [2], we add a normalization to teacher

Table 11. Ablation study on a large-scale dataset pre-trained with DAMO-YOLO-S. We validated our model on COCO without fine-tuning (denoted by '-'). For the VisDrone dataset, we loaded the pre-trained model and fine-tuned it for 300 epochs.

Pretrained Dataset	Downstream Task	AP
COCO	-	46.0
COCO+Obj365+OpenImage	**-**	**47.1**
COCO	VisDrone	24.6
COCO+Obj365+OpenImage	**VisDrone**	**26.6**

and student features, to weaken the effect the difference of real values brings. After subtracting the mean, standard deviation of each channel would function as temperature coefficient in KL loss.

Besides, we present two key observations for a better usage of distillation. One is the balance between distillation and task loss. As shown in Fig.4, when we focus more on distillation (weight=10), the classification loss in student has a slow convergence, which results in a negative effect. The small loss weight[3] (weight=0.5) hence is necessary to strike a balance between distillation and classification. The other is the shallow head of detector. We found that the proper reduction of the head depth is beneficial to the feature distillation on neck. The reason is that when the gap between the final outputs and the distilled feature map is closer, distillation can have a better impact on decision.

In practice, we use DAMO-YOLO-S as teacher to distill DAMO-YOLO-T, and DAMO-YOLO-M as teacher to distill DAMO-YOLO-S, and DAMO-YOLO-L as teacher to distill DAMO-YOLO-M, while DAMO-YOLO-L is distilled by it self.

2.5. General Class DAMO-YOLO Model

In this section, we introduce the General Class DAMO-YOLO model, which has been trained on multiple large-scale datasets, including COCO, Objects365, and OpenImage. Our model incorporates several improvements that enable high precision and generalization. Firstly, we address ambiguity resulting from overlapping categories across different datasets, such as "mouse," which can refer to a computer mouse in COCO/Objects365 and a rodent in OpenImage, by creating a unified label space for filtering. This reduces the original 945 categories (80 from COCO, 365 from Objects365, and 500 from OpenImage) to 701 [42]. Secondly, we present a method of polynomial smoothing followed by weighted sampling to address the imbalance in dataset sizes, which often results in batch sampling biases towards larger datasets. Finally, we implement a class-aware sampling approach to tackle the issue of long-tail effects in Objects365 and OpenImage datasets. This assigns greater sampling weights to images containing categories with fewer data points.

By improving our model, we trained DAMO-YOLO-S on Large-Scale Datasets and achieved a 1.1% mAP improvement compared to the DAMO-YOLO-S baseline on COCO. Furthermore, this pre-trained model can be fine-tuned for downstream tasks. We applied it to the VisDrone dataset for 300 epochs and achieved a 26.6% mAP, surpassing the performance of the model pre-trained on COCO. These results highlight the advantages of large-scale dataset training and the robustness it provides to the model. The results are presented in Table.11.

3. Implementation Details

Our models are trained 300 epochs with SGD optimizer. The weight decay and SGD momentum are 5e-4 and 0.9 respectively. The initial learning rate is 0.4 with a batch size of 256, and the learning rate decays according to a cosine schedule. Following YOLO-Series [9, 18, 32, 34] model exponential moving average (EMA) and grouped weight decay are used. To enhance data diversity, Mosaic [1, 32] and Mixup [41] augmentation is a common practice. However, recent advancement [3, 44] shows that properly designed box-level augmentation is crucial in object detection. Inspired by this, we apply Mosaic and Mixup for image-level augmentation and employ the box-level augmentation of SADA [3] after image-level augmentation for more robust augmentation.

4. Comparison with the SOTA

DAMO-YOLO release a series of general models and lightweight models, catering to both general scenarios and resource-limited edge scenarios.

In order to evaluate the performance of DAMO-YOLO's general models against other state-of-the-art models, we conducted a comparative analysis on T4-GPU using the TensorRT-FP16 inference engine. The results, presented in Table.9, demonstrate that DAMO-YOLO's general models outperform its rivals in terms of both accuracy and speed. Moreover, our proposed distillation technique offers significant improvements in accuracy.

To assess the performance of light weight models, we conducted a comparative analysis on Intel-8163 CPU using Openvino as the inference engine. As shown in Table 2, DAMO-YOLO's lightweight model achieved substantial leading advantages, surpassing its competitors by a considerable margin in both speed and accuracy.

[3] In our method, the cosine weight is utilized.

5. Conclusion

In this paper, we propose a new object detection method called DAMO-YOLO, the performance of which is superior to other methods in YOLO series. Its advantages come from new techs, including MAE-NAS backbone, efficient RepGFPN neck, ZeroHead, AlignedOTA label assignment and distillation enhancement.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. 2, 8

[2] Weihan Cao, Yifan Zhang, Jianfei Gao, Anda Cheng, Ke Cheng, and Jian Cheng. Pkd: General distillation framework for object detectors via pearson correlation coefficient. *arXiv preprint arXiv:2207.02039*, 2022. 7

[3] Yukang Chen, Yanwei Li, Tao Kong, Lu Qi, Ruihang Chu, Lei Li, and Jiaya Jia. Scale-aware automatic augmentation for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9563–9572, 2021. 8

[4] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. Detnas: Backbone search for object detection. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[5] Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Kaiqi Huang, Jungong Han, and Guiguang Ding. Re-parameterizing your optimizers rather than architectures. *arXiv preprint arXiv:2205.15242*, 2022. 5

[6] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13733–13742, 2021. 5

[7] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3490–3499. IEEE Computer Society, 2021. 2, 6

[8] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 303–312, 2021. 2, 6

[9] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021, 2021. 2, 5, 8

[10] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2019. 2, 3

[11] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 6

[14] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 3

[15] Tao Huang, Yuan Zhang, Shan You, Fei Wang, Chen Qian, Jian Cao, and Chang Xu. Masked distillation with receptive tokens. *arXiv preprint arXiv:2205.14589*, 2022. 6

[16] Chenhan Jiang, Hang Xu, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Sp-nas: Serial-to-parallel backbone search for object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11863–11872, 2020. 2

[17] Yiqi Jiang, Zhiyu Tan, Junyan Wang, Xiuyu Sun, Ming Lin, and Hao Li. Giraffedet: A heavy-neck paradigm for object detection. In *International Conference on Learning Representations*, 2022. 2, 3

[18] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications, 2022. 2, 5, 8

[19] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 6356–6364, 2017. 6

[20] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *Advances in Neural Information Processing Systems*, 33:21002–21012, 2020. 5, 6

[21] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2, 6

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[24] 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)*, pages 779–788, Los Alamitos, CA, USA, jun 2016. IEEE Computer Society. 2

[25] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2016. 2

[26] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018. 2

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2

[28] Changyong Shu, Yifan Liu, Jianfei Gao, Zheng Yan, and Chunhua Shen. Channel-wise knowledge distillation for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5311–5320, 2021. 6

[29] Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, and Wanli Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 2

[30] Zhenhong Sun, Ming Lin, Xiuyu Sun, Zhiyu Tan, Hao Li, and Rong Jin. Mae-det: Revisiting maximum entropy principle in zero-shot nas for efficient object detection. In *International Conference on Machine Learning*, pages 20810–20826. PMLR, 2022. 2, 3

[31] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020. 2, 3

[32] ultralytics. yolov5. https://github.com/ultralytics/yolov5, 2021. 2, 5, 8

[33] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-YOLOv4: Scaling cross stage partial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13029–13038, June 2021. 2

[34] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, 2022. 2, 5, 8

[35] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391, 2020. 3

[36] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9197–9206, 2019. 2, 3

[37] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4933–4942, 2019. 7

[38] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, and Baohua Lai. Pp-yoloe: An evolved version of yolo, 2022. 2, 5

[39] Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022. 6

[40] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 516–520, 2016. 6

[41] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 8

[42] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple multi-dataset detection. In *CVPR*, 2022. 8

[43] Benjin Zhu, Jianfeng Wang, Zhengkai Jiang, Fuhang Zong, Songtao Liu, Zeming Li, and Jian Sun. Autoassign: Differentiable label assignment for dense object detection. *arXiv preprint arXiv:2007.03496*, 2020. 2, 6

[44] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. In *European conference on computer vision*, pages 566–583. Springer, 2020. 8