美团 · YOLOv6

By Meituan Vision AI Department

YOLOV6

VISUAL LEARNERS

美团 · YOLOv6

By Meituan Vision AI Department

YOLOV6

VISUAL LEARNERS


VISUAL LEARNERS

WHAT IS YOLOV6?

The YOLOv6 paper was published by researchers at Meituan.

- The initial codebase of YOLOv6 was released in June 2022.
- It is perhaps the BEST and most improved version of the YOLO models.
- The first paper, along with the updated versions of the model (v2) was published in September.
- Eventhough it's paper came late than YOLOv7, It is considered the most accurate of all object detectors.

HOLD ON, WHAT IS MEITUAN?

- **Meituan (Chinese: 美团),** formerly Meituan-Dianping, is a Chinese shopping platform for locally found consumer products and retail services including entertainment, dining, delivery, travel and other services.

- The company is headquartered in Beijing and was founded in **2010** by **Wang Xing**.

- YOLOv6 was released by **Meituan's Vision AI department**.

HOW DOES YOLOV6 WORK?

Anchor free

It provides better generalizability and costs less time in post-processing.

Improvements

Longer training epochs, quantization, and knowledge distillation.

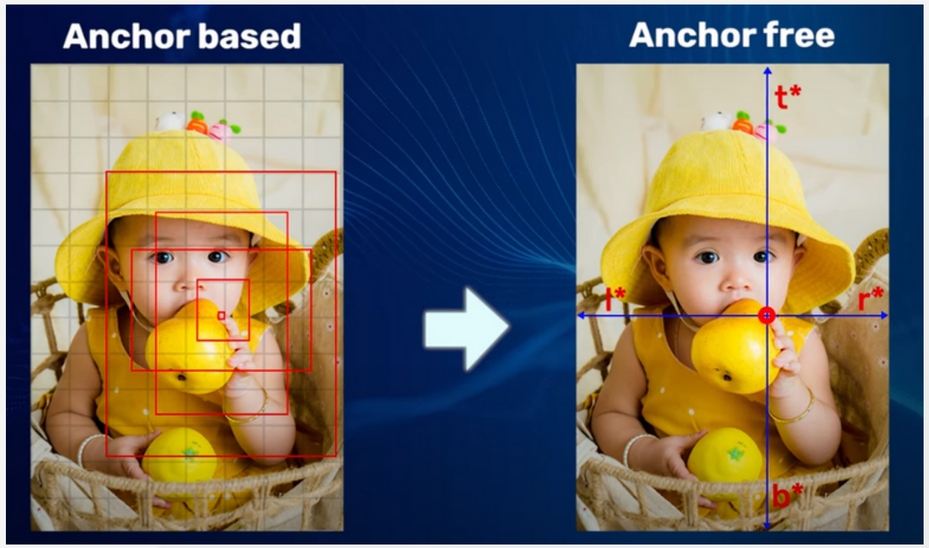
Architecture

YOLOv6 comes with a revised reparameterized backbone and neck.

Loss functions

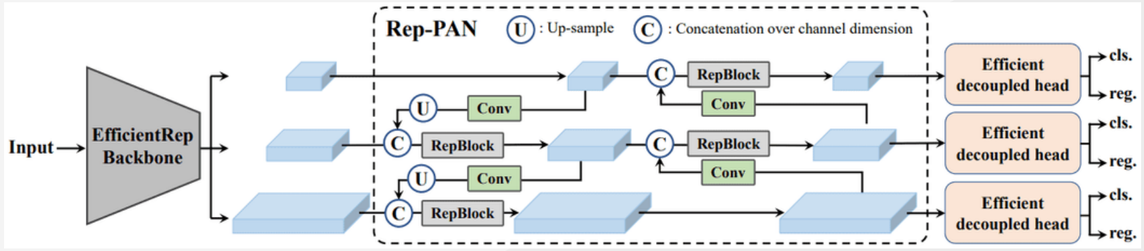
It used Varifocal loss (VFL) for classification and Distribution Focal loss (DFL) for detection.

WHAT'S NEW IN YOLOV6?



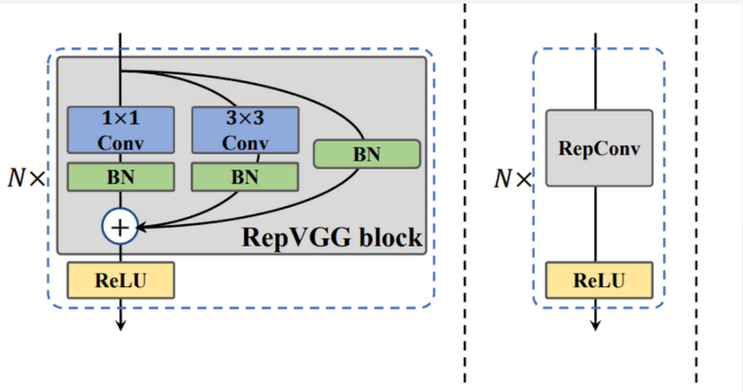
- Unlike the previous YOLO architectures, which use anchor-based methods for object detection, YOLOv6 opts for the anchor-free method.
- This makes YOLOv6 51% faster compared to most anchor-based object detectors. This is possible because it has 3 times fewer predefined priors.
- YOLOv6 uses the EfficientRep backbone consisting of RepBlock, RepConv, and CSPStackRep blocks.
- Further, YOLOv6 uses VFL and DFL as loss functions for classification and box regression, respectively.

YOLOV6 MODEL ARCHITECTURE



- Several modern and state-of-the-art practical techniques have been used to make all the YOLOv6 models as fast and accurate as possible.
- As with any other YOLO model, the YOLOV6 too has three components. They are the Backbone, the Neck, and the Head of the network, and all have something new to offer.
- As mentioned earlier, one of the biggest aspects of YOLOv6 is that it is anchor free and uses a reparameterized backbone!
- The image on the left is a complete display of the YOLOv6 object detection model architecture.

BACKBONE ARCHITECTURE



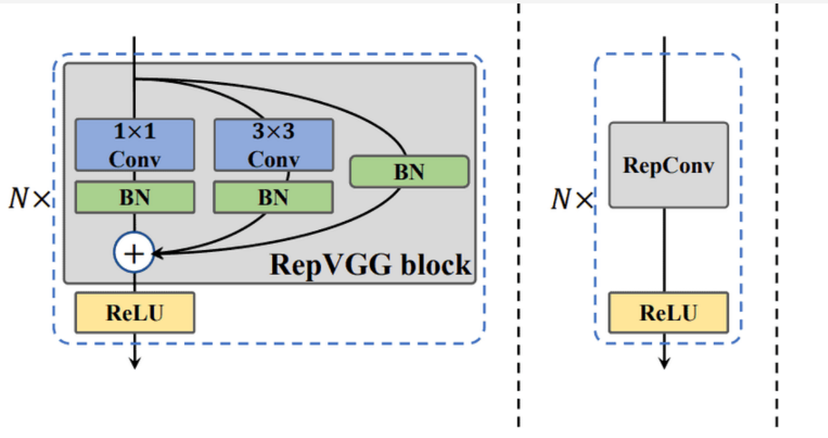
- Although multi-branch networks like ResNets provide better classification performance, they are slower during inference.

- Whereas, linear networks like VGG are much faster because of their effective 3×3 convolutions. However, they do not reach as high an accuracy as ResNets or networks with residual connections.

- For this reason, the YOLOv6 models use reparameterized backbones. In reparameterization, the network structure changes during training and inference

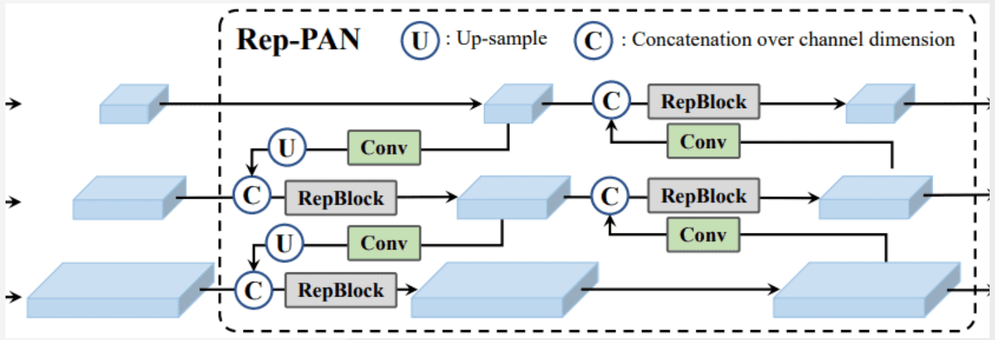
- RepVGG blocks are used while training YOLOv6, while RepConv blocks are used during inference.

BACKBONE ARCHITECTURE



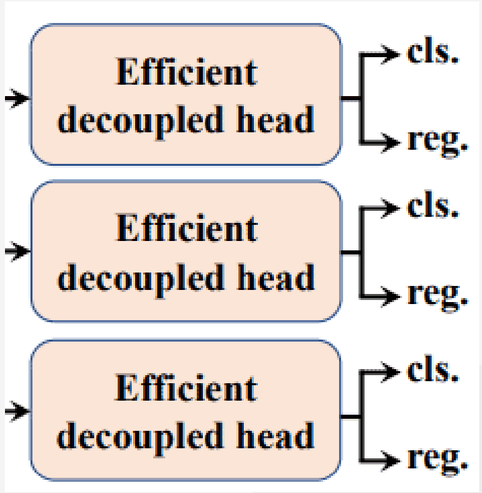
- The image on the left shows the reparameterized VGG (RepBlock) block with skip connections. This is used during the training phase of the YOLOv6 architectures.
- For the Medium and Large models, the YOLOv6 architecture uses reparameterized versions of the CSP backbone. We call it the CSPStackRep.
- **The entire backbone of the YOLOv6 architecture is called EfficientRep.**

NECK ARCHITECTURE



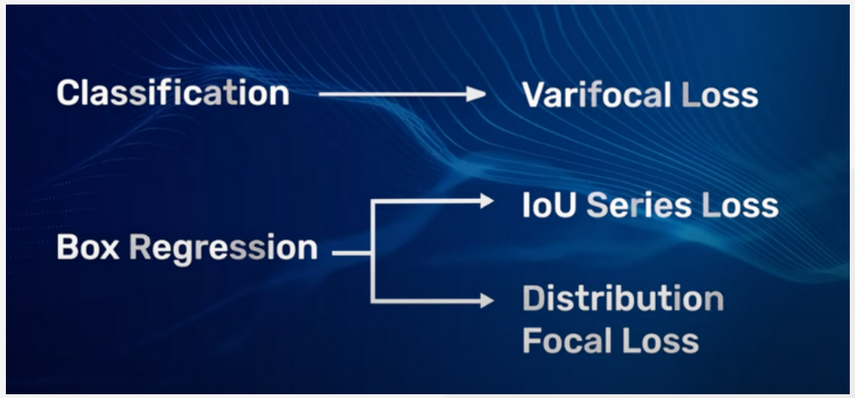
- In most object detection models, the neck aggregates the multi-scale feature maps using PAN (Path Aggregation Networks).
- This is not different in YOLOv6 and similar to what happens in YOLOv4 and YOLOv5.
- **The PAN in YOLOv6 concatenates features from various reparameterized blocks. For this reason, it is called reparameterized PAN or Rep-PAN for short.**

DETECTION HEAD



- Unlike YOLOv4 and YOLOv5, the YOLOv6 architecture used the Efficient Decoupled Head.
- This means that the classification and detection branches do not share the parameters and branch out from the backbone separately.
- **This further reduces computations and provides higher accuracy as well.**

LOSS FUNCTIONS IN YOLOV6



- The YOLOv6 object detection model requires two loss functions.

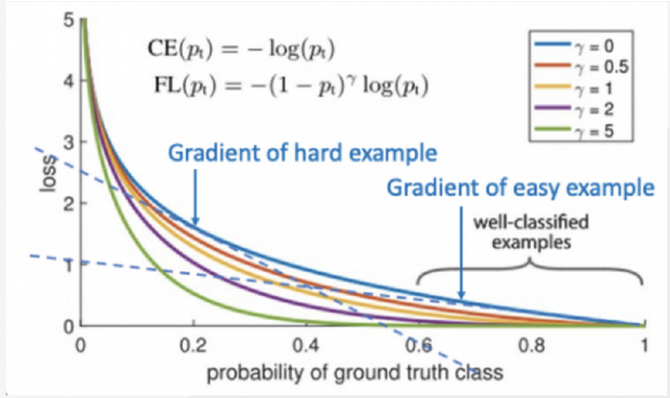
1. VFL (Varifocal Loss) as classification loss.
2. DFL (Distribution Focal Loss) along with SIoU or GIoU as box regression loss

- One is for classification and the other is for localization. We can call them classification loss and box regression loss.

VARIFOCAL LOSS FOR CLASSIFICATION

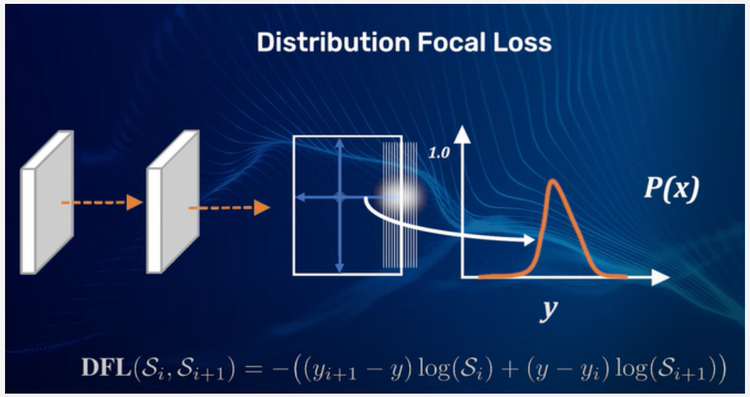
Varifocal Loss: $$\mathrm{VFL}(p,q)=\begin{cases}-q(q\log(p)+(1-q)\log(1-p)) & q>0\\ -\alpha p^{\gamma}\log(1-p) & q=0\end{cases}$$

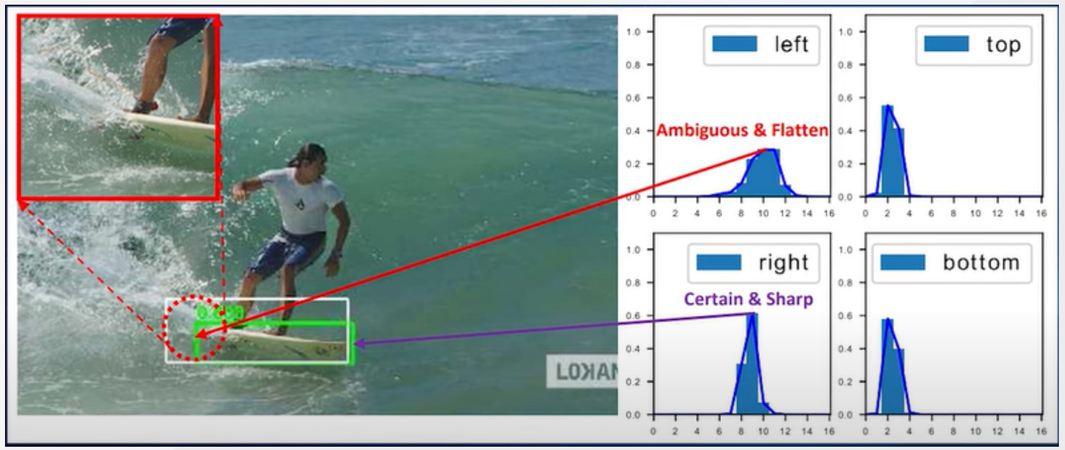
Focal Loss: $$\mathrm{FL}(p,y)=\begin{cases}-\alpha(1-p)^{\gamma}\log(p) & \text{if } y=1\\ -(1-\alpha)p^{\gamma}\log(1-p) & \text{otherwise}\end{cases}$$



- VFL originates from focal loss. This means it already takes care of hard and easy examples during training and weighs them differently.
- In addition, VFL also treats the positive and negative examples at different degrees of importance. This helps in balancing the learning signals from both samples.
- The varifocal loss treats the positive examples and the negative examples asymmetrically
- Unlike the focal loss which treats them equally. This means that the loss doesn't decrease for positive examples by the same amount that it decreases for negative examples.

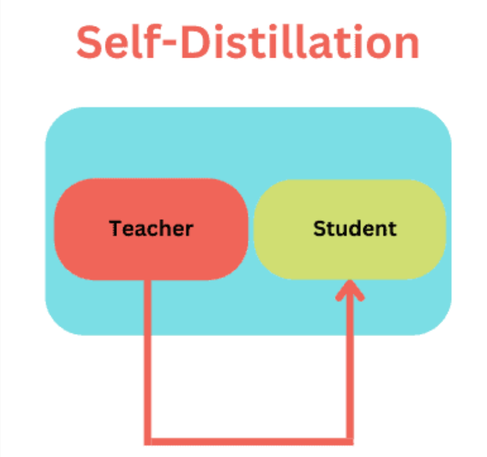
DISTRIBUTION FOCAL LOSS FOR BOX REGRESSION





- YOLOv6 Medium and Large models use DFL for box regression loss. DFL treats the continuous distribution of box locations as a discretized probability distribution.
- It is especially helpful in detection when the boundaries of the ground truth are blurred. DFLv2 was also experimented with, which introduced a lightweight sub-network.
- But this also meant extra computations, and no improvements over DFL were observed. So, they stuck to DFL as the localization loss function.

YOLOV6 IMPROVEMENTS FOR INDUSTRIAL APPLICATIONS



Longer Training

- Few YOLOv6 models were trained for 400 epochs instead of the general 300 epochs. This led to a better convergence.

Self-Distillation

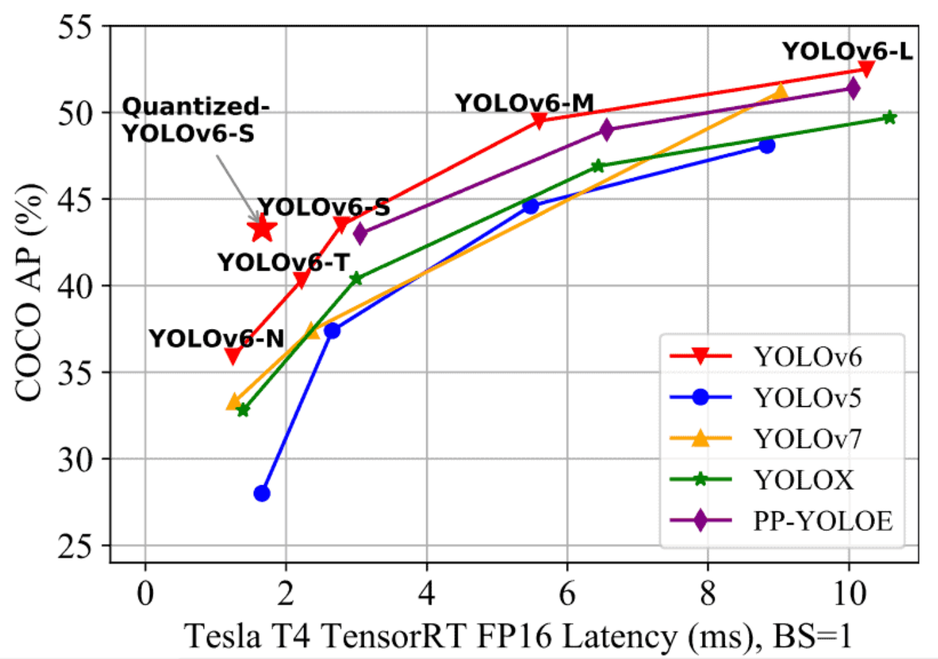
- YOLOv6 uses knowledge distillation to further improve the accuracy of the models. This is possible without involving a huge computation cost as well.
- In knowledge distillation, a teacher model is used to train a student model. The predictions of the teacher model act as soft labels along with the ground truth to train the student model.

TRAINING IMPLEMENTATION DETAILS

- The following table shows the hyperparameters, and hardware details for training the YOLOv6 model.

Optimizer	Stochastic Gradient Descent (SGD) with momentum and cosine decay learning rate
Optimizer Weight Decay	Exponential Moving Average (EMA)
Data Augmentation	Mosaic and Mixup
Training Data	COCO 2017 training set
Validation Data	COCO 2017 validation set
Training Hardware	8 NVIDIA A100 GPUs
Speed Benchmark Hardware	NVIDIA Tesla T4 GPU with TensorRT version 7.2

COCO BENCHMARK AND COMPARISON



- The following figure shows the comparison of mAP and latency between YOLOv6 and other models.
- One important observation that can be gleaned from the above figure is that the Quantized YOLOv6-S model is faster and has higher mAP compared to its counterparts in other YOLO versions.
- In terms of mAP, all the YOLOv6 models seem to be performing better than the other YOLO versions.

COCO BENCHMARK AND COMPARISON

Method	Input Size	AP^{val}	AP^{val}_{50}	FPS (bs=1)	FPS (bs=32)	Latency (bs=1)	Params	FLOPs
YOLOv5-N [10]	640	28.0%	45.7%	602	735	1.7 ms	1.9 M	4.5 G
YOLOv5-S [10]	640	37.4%	56.8%	376	444	2.7 ms	7.2 M	16.5 G
YOLOv5-M [10]	640	45.4%	64.1%	182	209	5.5 ms	21.2 M	49.0 G
YOLOv5-L [10]	640	49.0%	67.3%	113	126	8.8 ms	46.5 M	109.1 G
YOLOX-Tiny [7]	416	32.8%	50.3%*	717	1143	1.4 ms	5.1 M	6.5 G
YOLOX-S [7]	640	40.5%	59.3%*	333	396	3.0 ms	9.0 M	26.8 G
YOLOX-M [7]	640	46.9%	65.6%*	155	179	6.4 ms	25.3 M	73.8 G
YOLOX-L [7]	640	49.7%	68.0%*	94	103	10.6 ms	54.2 M	155.6 G
PPYOLOE-S [45]	640	43.1%	59.6%	327	419	3.1 ms	7.9 M	17.4 G
PPYOLOE-M [45]	640	49.0%	65.9%	152	189	6.6 ms	23.4 M	49.9 G
PPYOLOE-L [45]	640	51.4%	68.6%	101	127	10.1 ms	52.2 M	110.1 G
YOLOv7-Tiny [42]	416	33.3%*	49.9%*	787	1196	1.3 ms	6.2 M	5.8 G
YOLOv7-Tiny [42]	640	37.4%*	55.2%*	424	519	2.4 ms	6.2 M	13.7 G*
YOLOv7 [42]	640	51.2%	69.7%	110	122	9.0 ms	36.9 M	104.7 G
YOLOv6-N	640	35.9%	51.2%	802	1234	1.2 ms	4.3 M	11.1 G
YOLOv6-T	640	40.3%	56.6%	449	659	2.2 ms	15.0 M	36.7 G
YOLOv6-S	640	43.5%	60.4%	358	495	2.8 ms	17.2 M	44.2 G
YOLOv6-M‡	640	49.5%	66.8%	179	233	5.6 ms	34.3 M	82.2 G
YOLOv6-L-ReLU‡	640	51.7%	69.2%	113	149	8.8 ms	58.5 M	144.0 G
YOLOv6-L‡	640	52.5%	70.0%	98	121	10.2 ms	58.5 M	144.0 G

- Let’s now look at the quantitative results of COCO 2017 validation set benchmarks.
- It is as clear as glass that the YOLOv6 models are performing better than the other YOLO models.
- It is even more astonishing that the YOLOv6-L-ReLU model with 58.5 million parameters is surpassing PPYOLOE-L and YOLOX-L models in both speed and accuracy.

LABEL ASSIGNMENT STRATEGIES

Method	AP^{val}
ATSS [51]	32.5%
SimOTA [7]	34.5%
TAL [5]	**35.0%**
DW [18]	33.4%
ObjectBox [48]	30.1%

Table 6: Comparisons of label assignment methods.

- Let's now look at the quantitative results of COCO 2017 validation set benchmarks.
- SimOTA, although successful and even used in YOLOX, is slower. Moreover, TAL provides 0.5% compared to SimOTA.
- Finally, all the YOLOv6 models use the TAL strategy for label assignment.

THANK YOU

VISUAL LEARNERS

Ring that notification bell to get latest updates and subscribe to my channel.