YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information

Chien-Yao Wang[1,2], I-Hau Yeh[2], and Hong-Yuan Mark Liao[1,2,3]

[1] Institute of Information Science, Academia Sinica, Taiwan
[2] National Taipei University of Technology, Taiwan
[3] Department of Information and Computer Engineering, Chung Yuan Christian University, Taiwan
kinyiu@iis.sinica.edu.tw, ihyeh@emc.com.tw, liao@iis.sinica.edu.tw

Abstract. Today's deep learning methods focus on how to design the objective functions to make the prediction as close as possible to the target. Meanwhile, an appropriate neural network architecture has to be designed. Existing methods ignore a fact that when input data undergoes layer-by-layer feature transformation, large amount of information will be lost. This paper delve into the important issues of information bottleneck and reversible functions. We proposed the concept of programmable gradient information (PGI) to cope with the various changes required by deep networks to achieve multiple objectives. PGI can provide complete input information for the target task to calculate objective function, so that reliable gradient information can be obtained to update network parameters. In addition, a lightweight network architecture – Generalized Efficient Layer Aggregation Network (GELAN) is designed. GELAN confirms that PGI has gained superior results on lightweight models. We verified the proposed GELAN and PGI on MS COCO object detection dataset. The results show that GELAN only uses conventional convolution operators to achieve better parameter utilization than the state-of-the-art methods developed based on depth-wise convolution. PGI can be used for variety of models from lightweight to large. It can be used to obtain complete information, so that train-from-scratch models can achieve better results than state-of-the-art models pre-trained using large datasets, the comparison results are shown in Figure 1. The source codes are released at https://github.com/WongKinYiu/yolov9.

Keywords: object detection · information bottleneck · reversible model

1 Introduction

In recent years, researchers in the field of deep learning have mainly focused on how to develop more powerful neural network architectures and learning methods, such as CNNs [21–23, 42, 55, 70, 71], Transformers [8, 9, 40, 41, 60, 68, 69], Perceivers [26, 26, 32, 52, 56, 80, 80], and Mambas [17, 39, 79]. In addition, some researchers have tried to develop more general objective functions, such as loss function [5, 45, 46, 50, 76, 77], label assignment [10, 12, 33, 66, 78] and auxiliary

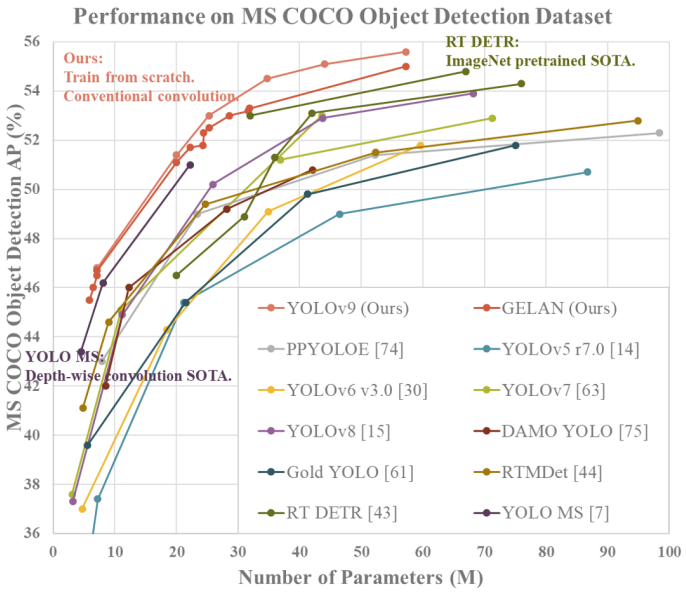

Fig. 1: Comparisons of the real-time object detectors on MS COCO dataset. The proposed GELAN and YOLOv9 surpassed all previous train-from-scratch methods in terms of object detection performance. In terms of accuracy, the proposed method outperforms RT DETR [43] pre-trained with a large dataset, and it also outperforms depth-wise convolution-based design YOLO MS [7] in terms of parameters utilization.

supervision [18, 20, 24, 28, 29, 51, 54, 67, 75]. The above studies all try to precisely find the mapping between input and target. However, most of past approaches have ignored that input data may have a non-negligible amount of information loss during the feedforward process, as shown in Figure 2. This loss of information can lead to biased learning process of the model. The above problems can result in DNNs to establish incorrect associations between targets and inputs, causing the trained model to produce untrustworthy predictions.

In DNNs, the phenomenon of information loss during the feedforward process is commonly known as information bottleneck [59]. The main methods that can alleviate this phenomenon are as follows: (1) Reversible architectures [3, 16, 19]: it mainly uses repeated input data to maintain the information of the input data in an explicit way; (2) Masked modeling [1, 6, 9, 27, 70, 72]: it mainly uses reconstruction loss to maximize the mutual information between extracted features and the input data in an implicit way; and (3) Deep supervision [28, 51, 54, 67]: it uses shallow features that have not lost too much important information to pre-establish a mapping from features to targets to ensure that important information can be propagated to deeper layers. However, the above methods have different drawbacks. For example, a reversible architecture requires additional layers to combine repeatedly fed input data, which will significantly increase the inference cost. In addition, since the input data layer to the output layer cannot have a too deep path, this limitation will make it difficult to model high-order

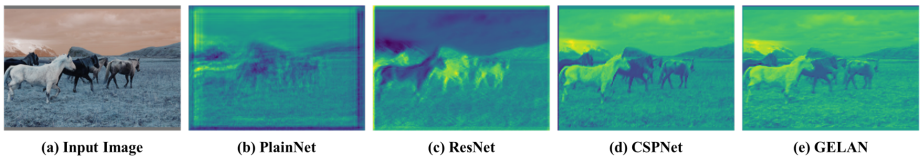

Fig. 2: Output feature visualization of different random initialized 50-layer networks: (a) input image, (b) PlainNet, (c) ResNet, (d) CSPNet, and (e) proposed GELAN.

semantic information during the training process. As for masked modeling, its reconstruction loss sometimes conflicts with the target loss. In addition, most of mask mechanisms also produce incorrect associations with data. For the deep supervision mechanism, it will produce error accumulation, and if the shallow supervision loses information during the training process, the subsequent layers will not be able to retrieve the required information. The above phenomenon will be more significant on difficult tasks and small models.

To address the above-mentioned issues, we propose a new concept, which is programmable gradient information (PGI). The concept is to generate reliable gradients through auxiliary reversible branch, so that the deep features can still maintain key characteristics for executing target task. The design of auxiliary reversible branch use multi-path features integration to avoid the semantic loss caused by a traditional deep supervision. In other words, we are programming gradient information propagation at different semantic levels, and thereby achieving better training results. The reversible architecture of PGI is built on auxiliary branch, so there is no additional inference cost. Since PGI can freely select loss function suitable for the target task, it also overcomes the problems encountered by masked modeling. The proposed PGI mechanism can be applied to various DNNs and is more general to various tasks.

In this paper, we also designed generalized ELAN (GELAN) based on ELAN [65]. GELAN simultaneously takes into account the parameter utilization, computational complexity, learning capability and inference speed. This design allows users to arbitrarily choose appropriate computational blocks for different inference devices. We combined the proposed PGI and GELAN to design YOLOv9. The experimental results verified that the proposed YOLOv9 achieved the top performance in all comparisons. Contributions of this paper are as follows:

1. We theoretically analyzed the existing DNNs from the perspective of reversible function, and successfully explained many phenomena that were difficult to explain in the past. We also designed PGI and auxiliary reversible branch based on this analysis and achieved excellent results.
2. The proposed PGI solves the important issues of existing reversible architectures and deep supervision methods, and shows strong versatility on various models and tasks.
3. The proposed GELAN shows great advantages of being light, fast, and accurate when comparing with the most advanced depth-wise convolution-based and Transformer-based model.
4. The performance of the proposed YOLOv9 on MS COCO dataset greatly surpasses the existing real-time object detectors in all aspects.

2 Related work

2.1 Real-time Object Detectors

The current mainstream real-time object detectors are the YOLO series [2, 7, 13–15, 25, 30, 31, 47–49, 61–63, 73, 74], and most of these models use CSPNet [64] or ELAN [65] and their variants as the main computational units. In terms of feature integration, improved PAN [37] or FPN [35] is often used, and then improved YOLOv3 head [49] or FCOS head [57, 58] is used as prediction head. Recently some real-time object detectors, such as RT DETR [43], which puts its foundation on DETR [4], have also been proposed. However, since it is extremely difficult for DETR-based object detector to be applied to new domains without a corresponding domain pre-trained model, the most widely used real-time object detector at present is still YOLO series. This paper proposed YOLOv9 which uses GELAN and PGI to improve the architecture and the training process.

2.2 Reversible Architectures

The operation unit of reversible architectures [3, 16, 19] must maintain the characteristics of reversible conversion, so it can be ensured that the output feature map of each layer of operation unit can retain complete original information. Before, RevCol [3] generalizes traditional reversible unit to multiple levels, and in doing so can expand the semantic levels expressed by different layer units. Through a literature review of various DNNs, we found that there are many high-performing architectures with varying degree of reversible properties. For example, Res2Net module [11] combines different input partitions with the next partition in a hierarchical manner, and concatenates all converted partitions before passing them backwards. CBNet [34, 38] re-introduces the original input data through composite backbone to obtain complete original information, and obtains different levels of multi-level reversible information through various composition methods. These network architectures generally have excellent parameter utilization, but the extra composite layers cause slow inference speeds. DynamicDet [36] combines CBNet [34] and YOLOv7 [63] to achieve a very good trade-off among speed, number of parameters, and accuracy. This paper designs reversible branches based on DynamicDet. In addition, reversible information is further introduced into the proposed PGI. The proposed new architecture does not require additional connections during the inference process, so it can fully retain the advantages of speed, number of parameters, and accuracy.

2.3 Auxiliary Supervision

Deep supervision [28, 54, 67] is the most common auxiliary supervision method, which performs training by inserting additional prediction layers in the intermediate layers. Especially the application of multi-layer decoders introduced in the transformer-based methods is the most common one. Another common auxiliary

supervision method is to utilize the relevant meta information to guide the intermediate layers to learn required properties of the target tasks [18, 20, 24, 29, 75]. Recently, there are many reports in the literature [53, 66, 81] that use different label assignment methods to generate different auxiliary supervision mechanisms to speed up the convergence speed of the model and improve the robustness at the same time. However, the auxiliary supervision mechanism is usually only applicable to large models, so when it is applied to lightweight models, it is easy to make the performance worse. The PGI we proposed designed a way to reprogram multi-level semantic information, and this design allows both large models and lightweight models to benefit from the auxiliary supervision mechanism.

3 Problem Statement

Usually, people attribute the difficulty of deep neural network convergence problem due to factors such as gradient vanish or gradient saturation, and these phenomena do exist in traditional deep neural networks. However, modern deep neural networks have already fundamentally solved the above problem by designing various normalization and activation functions. Nevertheless, deep neural networks still have the problem of slow convergence or poor convergence results.

In this paper, we explore the nature of the above issue further. Through in-depth analysis of information bottleneck, we deduced that the root cause of this problem is that the initial gradient originally coming from a very deep network has lost a lot of information needed to achieve the goal soon after it is transmitted. In order to confirm this inference, we feedforward deep networks of different architectures with random initial weights, and then illustrate them in Figure 2. Obviously, PlainNet has lost a lot of important information required for object detection in deep layers. As for the proportion of important information that ResNet, CSPNet, and GELAN can retain, it is indeed positively related to the accuracy that can be obtained after training. We further design reversible network-based methods to solve the above problems. In this section we elaborate our analysis of information bottleneck principle and reversible functions.

3.1 Information Bottleneck Principle

According to information bottleneck principle, we know that data X may cause information loss when going through transformation, as shown in Eq. 1 below:

$$I(X, X) \geq I(X, f_\theta(X)) \geq I(X, g_\phi(f_\theta(X))), \tag{1}$$

where I indicates mutual information, f and g are transformation functions, and θ and ϕ are parameters of f and g, respectively.

In deep neural networks, $f_\theta(\cdot)$ and $g_\phi(\cdot)$ respectively represent the operations of two consecutive layers in deep neural network. From Eq. 1, we can predict that as the number of network layer becomes deeper, the original data will be more likely to be lost. However, the gradient for updating parameters of the

deep neural network are based on loss function calculated by the output of the network as well as the given target. As one can imagine, the output of a deeper neural network is less able to retain complete information about the prediction target. This will make it possible to use incomplete information during network training, resulting in unreliable gradients and poor convergence.

3.2 Reversible Functions

When a function r has an inverse transformation function v, we call this function reversible function, as shown in Eq. 2.

$$X = v_\zeta(r_\psi(X)), \tag{2}$$

where ψ and ζ are parameters of r and v, respectively. Data X is converted by reversible function without losing information, as shown in Eq. 3.

$$I(X, X) = I(X, r_\psi(X)) = I(X, v_\zeta(r_\psi(X))). \tag{3}$$

When the network's transformation function is composed of reversible functions, more reliable gradients can be obtained to update the model. Almost all of today's popular models conform to the reversible property, such as Eq. 4.

$$X^{l+1} = X^l + f_\theta^{l+1}(X^l), \tag{4}$$

where l indicates the l-th layer of a PreAct ResNet and f is the transformation function of the l-th layer. PreAct ResNet [22] repeatedly passes the original data X to subsequent layers in an explicit way. Although such a design can make a deep neural network with more than a thousand layers converge very well, it destroys an important reason why we need deep neural networks. That is, for difficult problems, it is difficult for us to directly find simple mapping functions to map data to targets. This also explains why PreAct ResNet performs worse than ResNet [21] when the number of layers is small.

In addition, masked modeling approaches use approximation methods to try to find the inverse transformation v of r, so that the transformed features can retain enough information using sparse features. The formula is as follows:

$$X = v_\zeta(r_\psi(X) \cdot M), \tag{5}$$

where M is a dynamic binary mask. Other methods that are commonly used to perform the above tasks are diffusion model and variational autoencoder, and they both have the function of finding the inverse function. However, when we apply the above approach to a lightweight model, there will be defects because the lightweight model will be under parameterized to a large amount of raw data. Because of the above reason, important information $I(Y, X)$ that maps data X to target Y will also face the same problem. For this issue, we will explore it using the concept of information bottleneck [59], which is formulated as follows:

$$I(X, X) \geq I(Y, X) \geq I(Y, f_\theta(X)) \geq ... \geq I(Y, \hat{Y}). \tag{6}$$

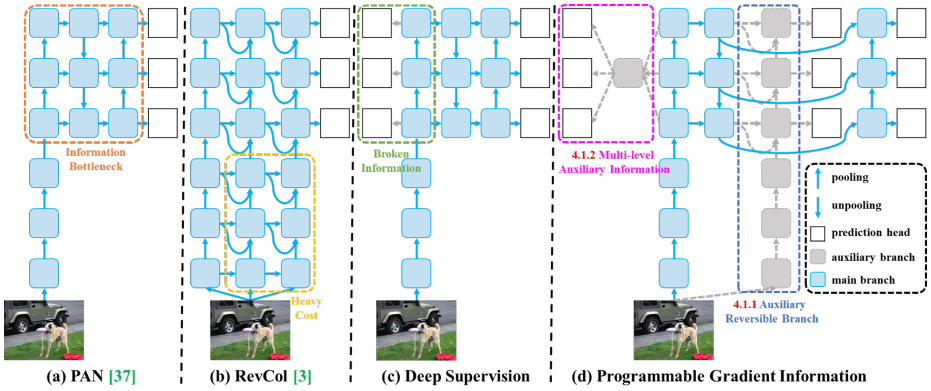

Fig. 3: PGI and related network architectures and methods. (a) Path Aggregation Network (PAN)) [37], (b) Reversible Columns (RevCol) [3], (c) conventional deep supervision, and (d) proposed Programmable Gradient Information (PGI). PGI is mainly composed of three components: (1) main branch, (2) auxiliary reversible branch, and (3) multi-level auxiliary information.

Generally speaking, $I(Y, X)$ will only occupy a very small part of $I(X, X)$. However, it is critical to the target task. Therefore, even if the amount of information lost in the feedforward stage is not significant, as long as $I(Y, X)$ is covered, the training effect will be greatly affected. The lightweight model itself is in an under parameterized state, so it is easy to lose a lot of important information in the feedforward stage. Therefore, our goal is to accurately filter $I(Y, X)$ from $I(X, X)$. As for fully preserving the information of X, that is difficult to achieve. Based on the above analysis, we hope to propose a training method that can not only generate reliable gradients to update the model, but also be suitable for shallow and lightweight neural networks.

4 Methodology

4.1 Programmable Gradient Information

In order to solve the aforementioned problems, we propose a new auxiliary supervision framework called Programmable Gradient Information (PGI), as shown in Figure 3 (d). PGI mainly includes three components, namely (1) main branch, (2) auxiliary reversible branch, and (3) multi-level auxiliary information. From Figure 3 (d) we see that the inference process of PGI only uses main branch and therefore does not require any additional inference cost. As for the other two components, they are used to solve or slow down several important issues in deep learning methods. Among them, auxiliary reversible branch is designed to deal with the problems caused by the deepening of neural networks. Network deepening will cause information bottleneck, which will make the loss function unable to generate reliable gradients. As for multi-level auxiliary information, it is designed to handle the error accumulation problem caused by deep supervision, especially for the lightweight model of multiple prediction branch. Next, we will introduce these two components step by step.

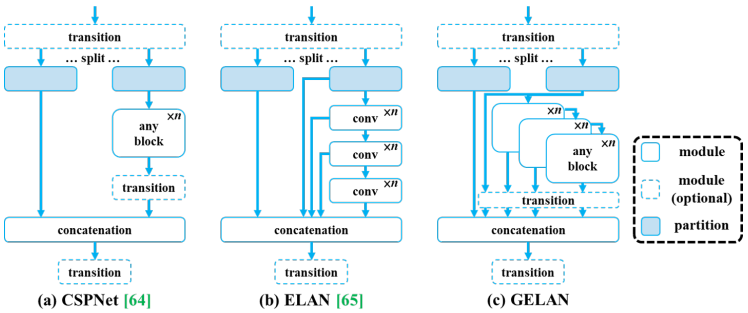

Fig. 4: The architecture of GELAN: (a) CSPNet [64], (b) ELAN [65], and (c) GELAN. GELAN imitates CSPNet and extend ELAN to support any computational blocks.

Auxiliary Reversible Branch In PGI, we propose auxiliary reversible branch to generate reliable gradients and update network parameters. PGI makes the feedforward features maintain complete information that maps from data to targets, it avoids the loss function finding false correlations from incomplete feedforward features that are less relevant to the target. However, adding reversible architecture to main branch will consume a lot of inference costs. We analyzed the architecture of Figure 3 (b) and found that when additional connections from deep to shallow layers are added, the inference time will increase by 20%. When we repeatedly add the input data to the high-resolution computing layer of the network (yellow box), the inference time even exceeds twice the time.

Since our goal is to use reversible architecture to obtain reliable gradients, reversible architecture is not necessary in the inference stage. We regard reversible branch as a deep supervision branch, and then design auxiliary reversible branch, as shown in Figure 3 (d). As for the main branch deep features that would suffer from information bottleneck problem, they will receive reliable gradient information from the auxiliary reversible branch. These gradient information will drive parameter learning to assist in extracting correct and important information, and enable the main branch to obtain features that are more effective for the target task. Moreover, the reversible architecture performs worse on shallow networks than on general networks because complex tasks require conversion in deeper networks. Our proposed method does not force the main branch to retain complete original information but updates it by generating useful gradient through the auxiliary supervision mechanism. The advantage of this design is that the proposed method can also be applied to shallower networks.

Multi-level Auxiliary Information In this section we will discuss how multi-level auxiliary information works. The deep supervision architecture including multiple prediction branch is shown in Figure 3 (c). For object detection, different layer in the feature pyramids used to perform different tasks, for example together they can detect objects of different sizes. Therefore, after connecting to the deep supervision branch, the shallow features will be guided to learn the features required for small object detection, and at this time the system will regard the positions of objects of other sizes as the background. The above deed will

cause the deep features to lose a lot of information needed to predict the target object. Regarding this issue, we believe that each layer in the feature pyramid needs to receive information about all target to make subsequent main branch can retain complete information to learn predictions for various targets.

The concept of multi-level auxiliary information is to insert an integration network between the feature pyramid hierarchy layers of auxiliary supervision and the main branch, and then uses it to combine returned gradients from different prediction heads, as shown in Figure 3 (d). Multi-level auxiliary information is then to aggregate the gradient information containing all targets, and pass it to the main branch and then update parameters. At this time, the characteristics of the main branch's feature pyramid hierarchy will not be dominated by some specific target's information. As a result, our method alleviates the broken information problem in deep supervision. In addition, any integrated network can be used in multi-level auxiliary information. Therefore, we can plan the required semantic levels to guide the learning of network architectures of different sizes.

4.2 Generalized ELAN

Generalized efficient layer aggregation network (GELAN) is combined with CSPNet [64] and ELAN [65]. It takes into account lightweight, inference speed, and accuracy. Its overall architecture is shown in Figure 4. We generalized the capability of ELAN [65], which originally only used stacking of convolutional layers, to a new architecture that can use any computational blocks.

5 Experiments

5.1 Experimental setup and implementation details

We verify the proposed method with MS COCO dataset, and follow settings of YOLOv7 AF [63] to conduct experimental results. All models are trained from scratch for 500 epochs. For implementation details, we built general and extended version of YOLOv9 based on YOLOv7 [63] and Dynamic YOLOv7 [36] respectively. In the design of the network architecture, we replaced ELAN [65] with GELAN using CSPNet blocks [64] with planned RepConv [63] as computational blocks. We also simplified downsampling module and optimized anchor-free prediction head. As for the auxiliary loss part of PGI, we follow YOLOv7's auxiliary head setting. Please see Appendix for more settings and details.

5.2 Comparison with state-of-the-arts

Table 1 lists comparison of our proposed YOLOv9 with other train-from-scratch real-time object detectors. For small and medium-sized model, YOLOv9 achieves better parameter utilization than YOLO MS which uses depth-wise convolution. For compact-sized model, YOLOv9-C has 42% less parameters and 22% less calculations than YOLOv7 AF, but achieves the same AP (53%). For large-sized model, YOLOv9-E has 16% less parameters, 27% less calculations than

Table 1: Comparison of state-of-the-art real-time object detectors.

Model	#Param.	FLOPs	$AP^{val}_{50:95}$	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv6-N v3.0 [30]	4.7	11.4	37.0	52.7	–	–	–	–
YOLOv6-S v3.0 [30]	18.5	45.3	44.3	61.2	–	–	–	–
YOLOv6-M v3.0 [30]	34.9	85.8	49.1	66.1	–	–	–	–
YOLOv6-L v3.0 [30]	59.6	150.7	51.8	69.2	–	–	–	–
YOLOv7 [63]	36.9	104.7	51.2	69.7	55.9	31.8	55.5	65.0
YOLOv7-X [63]	71.3	189.9	52.9	71.1	51.4	36.9	57.7	68.6
YOLOv7-N AF [63]	3.1	8.7	37.6	53.3	40.6	18.7	41.7	52.8
YOLOv7-S AF [63]	11.0	28.1	45.1	61.8	48.9	25.7	50.2	61.2
YOLOv7 AF [63]	43.6	130.5	53.0	70.2	57.5	35.8	58.7	68.9
YOLOv8-N [15]	3.2	8.7	37.3	52.6	–	–	–	–
YOLOv8-S [15]	11.2	28.6	44.9	61.8	–	–	–	–
YOLOv8-M [15]	25.9	78.9	50.2	67.2	–	–	–	–
YOLOv8-L [15]	43.7	165.2	52.9	69.8	57.5	35.3	58.3	69.8
YOLOv8-X [15]	68.2	257.8	53.9	71.0	58.7	35.7	59.3	70.7
DAMO YOLO-T [74]	8.5	18.1	42.0	58.0	45.2	23.0	46.1	58.5
DAMO YOLO-S [74]	12.3	37.8	46.0	61.9	49.5	25.9	50.6	62.5
DAMO YOLO-M [74]	28.2	61.8	49.2	65.5	53.0	29.7	53.1	66.1
DAMO YOLO-L [74]	42.1	97.3	50.8	67.5	55.5	33.2	55.7	66.6
Gold YOLO-N [61]	5.6	12.1	39.6	55.7	–	19.7	44.1	57.0
Gold YOLO-S [61]	21.5	46.0	45.4	62.5	–	25.3	50.2	62.6
Gold YOLO-M [61]	41.3	87.5	49.8	67.0	–	32.3	55.3	66.3
Gold YOLO-L [61]	75.1	151.7	51.8	68.9	–	34.1	57.4	68.2
YOLO MS-N [7]	4.5	17.4	43.4	60.4	47.6	23.7	48.3	60.3
YOLO MS-S [7]	8.1	31.2	46.2	63.7	50.5	26.9	50.5	63.0
YOLO MS [7]	22.2	80.2	51.0	68.6	55.7	33.1	56.1	66.5
YOLOv9-T (Ours)	2.0	7.7	38.3	53.1	41.3	18.6	42.3	54.7
YOLOv9-S (Ours)	7.1	26.4	46.8	63.4	50.7	26.6	56.0	64.5
YOLOv9-M (Ours)	20.0	76.3	51.4	68.1	56.1	33.6	57.0	68.0
YOLOv9-C (Ours)	25.3	102.1	53.0	70.2	57.8	36.2	58.5	69.3
YOLOv9-E (Ours)	57.3	189.0	55.6	72.8	60.6	40.2	61.0	71.4

YOLOv8-X, and has significant improvement of 1.7% AP. We also include ImageNet pretrained model to make the comparison in Figure 5. As for the parameter utilization in the deep model, YOLOv9 shows the huge advantages of using PGI, it requires only 66% of the parameters while maintaining the accuracy as RT DETR-X. As for the amount of computation, YOLOv9 is also very competitive. The above comparison results show that our proposed YOLOv9 has significantly improved in all aspects compared with existing methods.

5.3 Ablation Studies

Generalized ELAN For GELAN, we first do ablation studies for computational blocks. We used Res blocks [21], Dark blocks [49], and CSP blocks [64] to conduct experiments, respectively. Table 2 shows that after replacing Conv blocks in ELAN with different computational blocks, the model can maintain good performance. Users are indeed free to replace computational blocks and use them on their respective inference devices. Among different computational block replacements, CSP blocks perform particularly well. They not only reduce the amount of parameters and computation, but also improve AP by 0.7%. Therefore, we choose CSP-ELAN as the component unit of GELAN in YOLOv9.

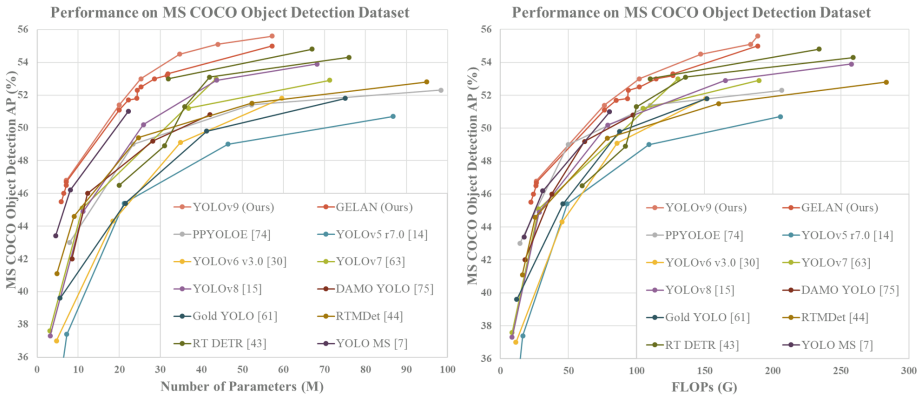

Fig. 5: Comparison of state-of-the-art real-time object detectors. The methods participating in the comparison all use ImageNet as pre-trained weights, including RT DETR [43], RTMDet [44], and PP-YOLOE [73], etc. The YOLOv9 that uses train-from-scratch method clearly surpasses the performance of other methods.

Table 2: Ablation study on various computational blocks.

Model	Computational Block	#Param.	FLOPs	$AP^{val}_{50:95}$
GELAN-S	Conv	6.2M	23.5G	44.8%
GELAN-S	Res [21]	5.4M	21.0G	44.3%
GELAN-S	Dark [49]	5.7M	21.8G	44.5%
GELAN-S	CSP [64]	5.9M	22.4G	45.5%

Next, we conduct ELAN block-depth and CSP block-depth experiments on GELAN, and display the results in Table 3. We can see when the depth is greater than or equal to 2, no matter it is improving the ELAN depth or the CSP depth, the number of parameters, the amount of computation, and the accuracy will always show a linear relationship. This means GELAN is not sensitive to the depth, so users can arbitrarily adjust the depth of GELAN, and have a model with stable performance. In Table 3, for YOLOv9-{S,M,C}, we set the pairing of the ELAN depth and the CSP depth to {{2, 3}, {2, 1}, {2, 1}}.

Table 3: Ablation study on ELAN and CSP depth.

Model	Depth_{ELAN}	Depth_{CSP}	#Param.	FLOPs	$AP^{val}_{50:95}$
GELAN-S	2	1	5.9M	22.4G	45.5%
GELAN-S	2	2	6.5M	24.4G	46.0%
GELAN-S	3	1	7.1M	26.3G	46.5%
GELAN-S	2	3	7.1M	26.4G	46.7%
GELAN-M	2	1	20.0M	76.3G	51.1%
GELAN-M	2	2	22.2M	85.1G	51.7%
GELAN-M	3	1	24.3M	93.5G	51.8%
GELAN-M	2	3	24.4M	94.0G	52.3%
GELAN-C	1	1	18.9M	77.5G	50.7%
GELAN-C	2	1	25.3M	102.1G	52.5%
GELAN-C	2	2	28.6M	114.4G	53.0%
GELAN-C	3	1	31.7M	126.8G	53.2%
GELAN-C	2	3	31.9M	126.7G	53.3%

Programmable Gradient Information In terms of PGI, we performed ablation studies on auxiliary reversible branch and multi-level auxiliary information on the backbone and neck, respectively. We designed auxiliary reversible branch called reversible composite network (RCN) to use DHLC [34] linkage to obtain multi-level reversible information. As for multi-level auxiliary information, we use FPN and PAN for ablation studies and the role of PFH is equivalent to the traditional deep supervision. The results are listed in Table 4. From Table 4, we can see that PFH is only effective in extremely deep models, while our proposed PGI can improve accuracy under different combinations. Especially when using RCN, we get stable and better results. We also apply the lead-head guided assignment (LHG) [63] to PGI, and achieved much better performance.

Table 4: Ablation study on PGI of backbone and neck.

Model	$\mathbf{G}_{backbone}$	$\mathbf{G}_{neck}$	$\mathbf{AP}^{val}_{50:95}$	$\mathbf{AP}^{val}_{S}$	$\mathbf{AP}^{val}_{M}$	$\mathbf{AP}^{val}_{L}$
GELAN-C	–	–	52.5%	35.8%	57.6%	**69.4%**
GELAN-C	PFH	–	52.5%	35.3%	58.1%	68.9%
GELAN-C	FPN	–	52.6%	35.3%	58.1%	68.9%
GELAN-C	–	RCN	52.7%	35.3%	58.4%	68.9%
GELAN-C	FPN	RCN	52.8%	35.8%	58.2%	69.1%
GELAN-C	RCN	–	**52.9%**	35.2%	**58.7%**	68.6%
GELAN-C	LHG-RCN	–	**53.0%**	**36.3%**	58.5%	69.1%
GELAN-E	–	–	55.0%	38.0%	60.6%	70.9%
GELAN-E	PFH	–	55.3%	38.3%	60.3%	71.6%
GELAN-E	FPN	–	**55.6%**	**40.2%**	61.0%	71.4%
GELAN-E	PAN	–	55.5%	39.0%	**61.1%**	71.5%
GELAN-E	FPN	RCN	**55.6%**	39.8%	60.9%	**71.9%**

We further make comparisons of PGI and deep supervision, these results are shown in Table 5. We can see deep supervision can only bring gains in extremely deep model. The proposed PGI can effectively handle problems such as information bottleneck and information broken, and can comprehensively improve the accuracy of models at all scales. The concept of PGI brings two valuable contributions. The first one is to make the auxiliary supervision method applicable to shallow models, while the second one is to make the deep model training process obtain more reliable gradients. These gradients enable deep models to use more accurate information to establish correct correlations between data and targets.

Table 5: Ablation study on PGI.

Model	$\mathbf{AP}^{val}_{50:95}$		$\mathbf{AP}^{val}_{50}$		$\mathbf{AP}^{val}_{75}$	
GELAN-S	46.7%		63.0%		**50.7%**	
+ **Deep Supervision**	46.5%	-0.2	62.9%	-0.1	50.5%	-0.2
+ **PGI**	**46.8%**	+0.1	**63.4%**	+0.4	**50.7%**	=
GELAN-M	51.1%		67.9%		55.7%	
+ **Deep Supervision**	51.2%	+0.1	**68.2%**	+0.3	55.7%	=
+ **PGI**	**51.4%**	+0.3	68.1%	+0.2	**56.1%**	+0.4
GELAN-C	52.5%		69.5%		57.3%	
+ **Deep Supervision**	52.5%	=	69.9%	+0.4	57.1%	-0.2
+ **PGI**	**53.0%**	+0.5	**70.3%**	+0.8	**57.8%**	+0.5
GELAN-E	55.0%		71.9%		60.0%	
+ **Deep Supervision**	55.3%	+0.3	72.3%	+0.4	60.2%	+0.2
+ **PGI**	**55.6%**	+0.6	**72.8%**	+0.9	**60.6%**	+0.6

Latency of Model We further compare the latency of different versions of YOLO. The results are shown in Table 6. From the results we can see that the proposed YOLOv9 is the fastest and the most accurate model. The YOLOv9-C is 10% faster than YOLOv7 AF and maintain same accuracy.

Table 6: Batch 1 latency on T4 GPUs using TensorRT engine in FP16.

	YOLOv6-L v3.0 [30]	YOLOv7 AF [63]	YOLOv8-L [15]	YOLOv9-C (Ours)
Latency (ms)	7.9	6.7	8.1	**6.1**
$AP^{val}_{50:95}$ (%)	$51.8/52.8^{distill}$	**53.0**	52.9	**53.0**

5.4 The Power of PGI

Visualization of PGI Figure 6 is used to explore and visualize the information bottleneck issues and show whether PGI can provide more reliable gradients during the training process. From the comparison, we can clearly see that PGI accurately and concisely captures the area containing objects. As for GELAN that does not use PGI, we found that it had divergence when detecting object boundaries, and it also produced unexpected responses in some background areas. This experiment confirms that PGI can indeed provide better gradients to update parameters and enable the feedforward stage of the main branch can effectively capture the correct relationship between the input data and the target.

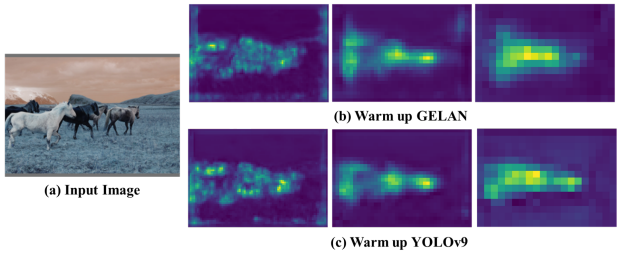

Fig. 6: PAN visualization of GELAN and YOLOv9 (GELAN + PGI) after one epoch of warm-up. GELAN originally had some divergence, but after adding PGI's reversible branch, it is more capable of focusing on the target object.

Versatility of PGI on Small Datasets Since PGI can more accurately capture the relation between data and targets, in theory it should be able to learn more useful information from smaller datasets. And the pre-trained YOLOv9 should have better transfer learning capabilities. Table 7 shows that PGI has excellent train-from-scratch and transfer learning capabilities on the small dataset.

Table 7: PGI on small datasets (VOC).

	GELAN-S	YOLOv9-S	GELAN-S	YOLOv9-S	YOLOv5-S	YOLOv8-S
pretrain	–	–	COCO	COCO	COCO	COCO
AP^{box}	64.4%	**65.1%**	73.5%	**74.4%**	62.4%	67.1%
AP^{box}_{50}	82.6%	**83.1%**	89.8%	**90.4%**	86.6%	85.8%

The Versatility of PGI in Various Tasks We extend PGI in different tasks, including instance segmentation, panoptic segmentation, and image captioning. In Table 8, the results prove that PGI can indeed be used in various tasks.

Table 8: PGI in various tasks.

	YOLOv9	YOLOv9 Segment	YOLOv9 Panoptic	YOLOv9 Caption
metric	AP^{box}	AP^{box}/AP^{mask}	$mIoU^{sem}/PQ^{pan}$	$BLEU4^{cap}$
no PGI	52.5%	52.3%/42.4%	39.0%/39.4%	38.8%
with PGI	**53.0%**	**52.9%/43.2%**	**39.8%/40.5%**	**39.1%**

The Versatility of PGI in Various Architectures We extend PGI in different training scheme and architectures, including mask-guided YOLOv9 (MG YOLOv9), light head YOLOv9 (LH YOLOv9), YOLOv9 with Transformer (YOLOv9 TR), YOLOv9 using hybrid convolution (YOLOv9 Lite), and YOLOv9 using depth-wise convolution (YOLOv9 Light). In Table 9, the results prove that PGI can indeed be used in various architectures.

Table 9: PGI in various models.

	MG YOLOv9	LH YOLOv9	YOLOv9 TR	YOLOv9 Lite	YOLOv9 Light
#param.	25.3M	21.1M	14.1M	13.3M	2.5M
FLOPs	102.1G	82.5G	67.5G	66.7G	11.0G
AP^{box}	53.3%	52.9%	53.1%	52.7%	44.1%

Training Cost of PGI Table 10 shows training time of YOLOv9-C with different PGI methods on RTX 6000 ada. The experimental results are shown in Table 10. It shows that using PGI will increase the training time by 30%∼40%.

Table 10: Training cost of PGI with batch size 128 training on RTX 6000 ada.

PGI method	–	**PFR**	**FPN**	**RCN**	**LHG-RCN**
min/epoch	7	8	9	11	10

6 Conclusions

In this paper, we propose to use PGI to solve the information bottleneck problem and the problem that the deep supervision mechanism is not suitable for lightweight neural networks. We designed GELAN, a highly efficient and flexible neural network for various devices. In terms of object detection, GELAN has strong and stable performance at different computational blocks and depth settings. It can indeed be widely expanded into a model suitable for various inference devices. For the above two issues, the introduction of PGI allows both lightweight models and deep models to achieve significant improvements in accuracy. The YOLOv9, designed by combining PGI and GELAN, has shown strong competitiveness. Its excellent design allows the deep model to reduce the number of parameters by 49% and the amount of calculations by 43% compared with YOLOv8, but it still has a 0.6% AP improvement on MS COCO dataset.

Acknowledgements

The authors wish to thank National Center for High-performance Computing (NCHC) for providing computational and storage resources.

References

1. Bao, H., Dong, L., Piao, S., Wei, F.: BEiT: BERT pre-training of image transformers. In: International Conference on Learning Representations (ICLR) (2022)
2. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
3. Cai, Y., Zhou, Y., Han, Q., Sun, J., Kong, X., Li, J., Zhang, X.: Reversible column networks. In: International Conference on Learning Representations (ICLR) (2023)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 213–229 (2020)
5. Chen, K., Lin, W., Li, J., See, J., Wang, J., Zou, J.: AP-loss for accurate one-stage object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **43**(11), 3782–3798 (2020)
6. Chen, Y., Liu, Y., Jiang, D., Zhang, X., Dai, W., Xiong, H., Tian, Q.: SdAE: Self-distillated masked autoencoder. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 108–124 (2022)
7. Chen, Y., Yuan, X., Wu, R., Wang, J., Hou, Q., Cheng, M.M.: YOLO-MS: rethinking multi-scale representation learning for real-time object detection. arXiv preprint arXiv:2308.05480 (2023)
8. Ding, M., Xiao, B., Codella, N., Luo, P., Wang, J., Yuan, L.: DaVIT: Dual attention vision transformers. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 74–92 (2022)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (ICLR) (2021)
10. Feng, C., Zhong, Y., Gao, Y., Scott, M.R., Huang, W.: TOOD: Task-aligned one-stage object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3490–3499 (2021)
11. Gao, S.H., Cheng, M.M., Zhao, K., Zhang, X.Y., Yang, M.H., Torr, P.: Res2Net: A new multi-scale backbone architecture. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **43**(2), 652–662 (2019)
12. Ge, Z., Liu, S., Li, Z., Yoshie, O., Sun, J.: OTA: Optimal transport assignment for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 303–312 (2021)
13. Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J.: YOLOX: Exceeding YOLO series in 2021. arXiv preprint arXiv:2107.08430 (2021)
14. Glenn, J.: YOLOv5 release v7.0. `https://github.com/ultralytics/yolov5/releases/tag/v7.0` (2022)
15. Glenn, J.: YOLOv8 release v8.1.0. `https://github.com/ultralytics/ultralytics/releases/tag/v8.1.0` (2024)
16. Gomez, A.N., Ren, M., Urtasun, R., Grosse, R.B.: The reversible residual network: Backpropagation without storing activations. Advances in Neural Information Processing Systems (NeurIPS) (2017)

17. Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752 (2023)
18. Guo, C., Fan, B., Zhang, Q., Xiang, S., Pan, C.: AugFPN: Improving multi-scale feature learning for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12595–12604 (2020)
19. Han, Q., Cai, Y., Zhang, X.: RevColV2: Exploring disentangled representations in masked image modeling. Advances in Neural Information Processing Systems (NeurIPS) (2023)
20. Hayder, Z., He, X., Salzmann, M.: Boundary-aware instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5696–5704 (2017)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
22. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 630–645. Springer (2016)
23. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4700–4708 (2017)
24. Huang, K.C., Wu, T.H., Su, H.T., Hsu, W.H.: MonoDTR: Monocular 3D object detection with depth-aware transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4012–4021 (2022)
25. Huang, L., Li, W., Shen, L., Fu, H., Xiao, X., Xiao, S.: YOLOCS: Object detection based on dense channel compression for feature spatial solidification. arXiv preprint arXiv:2305.04170 (2023)
26. Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., Carreira, J.: Perceiver: General perception with iterative attention. In: International Conference on Machine Learning (ICML). pp. 4651–4664 (2021)
27. Kenton, J.D.M.W.C., Toutanova, L.K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT. vol. 1, p. 2 (2019)
28. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: Artificial Intelligence and Statistics. pp. 562–570 (2015)
29. Levinshtein, A., Sereshkeh, A.R., Derpanis, K.: DATNet: Dense auxiliary tasks for object detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 1419–1427 (2020)
30. Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., Chu, X.: YOLOv6 v3.0: A full-scale reloading. arXiv preprint arXiv:2301.05586 (2023)
31. Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., et al.: YOLOv6: A single-stage object detection framework for industrial applications. arXiv preprint arXiv:2209.02976 (2022)
32. Li, H., Zhu, J., Jiang, X., Zhu, X., Li, H., Yuan, C., Wang, X., Qiao, Y., Wang, X., Wang, W., et al.: Uni-perceiver v2: A generalist model for large-scale vision and vision-language tasks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2691–2700 (2023)
33. Li, S., He, C., Li, R., Zhang, L.: A dual weighting label assignment scheme for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9387–9396 (2022)

34. Liang, T., Chu, X., Liu, Y., Wang, Y., Tang, Z., Chu, W., Chen, J., Ling, H.: CBNet: A composite backbone network architecture for object detection. IEEE Transactions on Image Processing (TIP) (2022)
35. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2117–2125 (2017)
36. Lin, Z., Wang, Y., Zhang, J., Chu, X.: DynamicDet: A unified dynamic architecture for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6282–6291 (2023)
37. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8759–8768 (2018)
38. Liu, Y., Wang, Y., Wang, S., Liang, T., Zhao, Q., Tang, Z., Ling, H.: CBNet: A novel composite backbone network architecture for object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). pp. 11653–11660 (2020)
39. Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Liu, Y.: Vmamba: Visual state space model. arXiv preprint arXiv:2401.10166 (2024)
40. Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al.: Swin transformer v2: Scaling up capacity and resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
41. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10012–10022 (2021)
42. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A ConvNet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11976–11986 (2022)
43. Lv, W., Xu, S., Zhao, Y., Wang, G., Wei, J., Cui, C., Du, Y., Dang, Q., Liu, Y.: DETRs beat YOLOs on real-time object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16965–16974 (2024)
44. Lyu, C., Zhang, W., Huang, H., Zhou, Y., Wang, Y., Liu, Y., Zhang, S., Chen, K.: RTMDet: An empirical study of designing real-time object detectors. arXiv preprint arXiv:2212.07784 (2022)
45. Oksuz, K., Cam, B.C., Akbas, E., Kalkan, S.: A ranking-based, balanced loss function unifying classification and localisation in object detection. Advances in Neural Information Processing Systems (NeurIPS) **33**, 15534–15545 (2020)
46. Oksuz, K., Cam, B.C., Akbas, E., Kalkan, S.: Rank & sort loss for object detection and instance segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3009–3018 (2021)
47. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)
48. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7263–7271 (2017)
49. Redmon, J., Farhadi, A.: YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)

50. Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 658–666 (2019)
51. Shen, Z., Liu, Z., Li, J., Jiang, Y.G., Chen, Y., Xue, X.: Object detection from scratch with deep supervision. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **42**(2), 398–412 (2019)
52. Shridhar, M., Manuelli, L., Fox, D.: Perceiver-actor: A multi-task transformer for robotic manipulation. In: Conference on Robot Learning (CoRL). pp. 785–799 (2023)
53. Sun, P., Jiang, Y., Xie, E., Shao, W., Yuan, Z., Wang, C., Luo, P.: What makes for end-to-end object detection? In: International Conference on Machine Learning (ICML). pp. 9934–9944 (2021)
54. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1–9 (2015)
55. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2818–2826 (2016)
56. Tang, Z., Cho, J., Lei, J., Bansal, M.: Perceiver-VL: Efficient vision-and-language modeling with iterative latent attention. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 4410–4420 (2023)
57. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9627–9636 (2019)
58. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: A simple and strong anchor-free object detector. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **44**(4), 1922–1933 (2022)
59. Tishby, N., Zaslavsky, N.: Deep learning and the information bottleneck principle. In: IEEE Information Theory Workshop (ITW). pp. 1–5 (2015)
60. Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., Li, Y.: MaxVIT: Multi-axis vision transformer. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 459–479 (2022)
61. Wang, C., He, W., Nie, Y., Guo, J., Liu, C., Han, K., Wang, Y.: Gold-YOLO: Efficient object detector via gather-and-distribute mechanism. Advances in Neural Information Processing Systems (NeurIPS) (2023)
62. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: Scaled-YOLOv4: Scaling cross stage partial network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13029–13038 (2021)
63. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7464–7475 (2023)
64. Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W., Yeh, I.H.: 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 (CVPRW). pp. 390–391 (2020)
65. Wang, C.Y., Liao, H.Y.M., Yeh, I.H.: Designing network design strategies through gradient path analysis. Journal of Information Science and Engineering (JISE) (2023)

66. Wang, J., Song, L., Li, Z., Sun, H., Sun, J., Zheng, N.: End-to-end object detection with fully convolutional network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15849–15858 (2021)
67. Wang, L., Lee, C.Y., Tu, Z., Lazebnik, S.: Training deeper convolutional networks with deep supervision. arXiv preprint arXiv:1505.02496 (2015)
68. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 568–578 (2021)
69. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: PVT v2: Improved baselines with pyramid vision transformer. Computational Visual Media **8**(3), 415–424 (2022)
70. Woo, S., Debnath, S., Hu, R., Chen, X., Liu, Z., Kweon, I.S., Xie, S.: ConvNeXt v2: Co-designing and scaling convnets with masked autoencoders. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16133–16142 (2023)
71. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1492–1500 (2017)
72. Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q., Hu, H.: SimMIM: A simple framework for masked image modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9653–9663 (2022)
73. Xu, S., Wang, X., Lv, W., Chang, Q., Cui, C., Deng, K., Wang, G., Dang, Q., Wei, S., Du, Y., et al.: PP-YOLOE: An evolved version of YOLO. arXiv preprint arXiv:2203.16250 (2022)
74. Xu, X., Jiang, Y., Chen, W., Huang, Y., Zhang, Y., Sun, X.: DAMO-YOLO: A report on real-time object detection design. arXiv preprint arXiv:2211.15444 (2022)
75. Zhang, R., Qiu, H., Wang, T., Guo, Z., Cui, Z., Qiao, Y., Li, H., Gao, P.: MonoDETR: Depth-guided transformer for monocular 3D object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9155–9166 (2023)
76. Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., Ren, D.: Distance-IoU loss: Faster and better learning for bounding box regression. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). vol. 34, pp. 12993–13000 (2020)
77. Zhou, D., Fang, J., Song, X., Guan, C., Yin, J., Dai, Y., Yang, R.: IoU loss for 2D/3D object detection. In: International Conference on 3D Vision (3DV). pp. 85–94 (2019)
78. Zhu, B., Wang, J., Jiang, Z., Zong, F., Liu, S., Li, Z., Sun, J.: AutoAssign: Differentiable label assignment for dense object detection. arXiv preprint arXiv:2007.03496 (2020)
79. Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., Wang, X.: Vision mamba: Efficient visual representation learning with bidirectional state space model. arXiv preprint arXiv:2401.09417 (2024)
80. Zhu, X., Zhu, J., Li, H., Wu, X., Li, H., Wang, X., Dai, J.: Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16804–16815 (2022)
81. Zong, Z., Song, G., Liu, Y.: DETRs with collaborative hybrid assignments training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6748–6758 (2023)