Journal of Computers Vol. 35 No. 3, June 2024, pp. 175-192
doi: 10.53106/199115992024063503013

DETRs with Dynamic Contrastive Denoising Training for Smartphone Assembly Parts

Hang Ma, Yu-Hang Zhang, Bo-Si Liu, Wen-Bai Chen*

School of automation, Beijing Information Science and Technology University, China
chenwb@bistu.edu.cn

Received 27 March 2024; Revised 20 May 2024; Accepted 23 May 2024

Abstract. In the scenario of 3C (Computer, Communication, Consumer Electronics), the algorithm for detecting targets in smartphone component assembly consumes a substantial amount of system computing resources.It also faces challenges such as the flexible nature of target components and the small scale of heterogeneous components, leading to low detection accuracy. To adapt to the 3C scenario, this paper proposes improvements based on the DINO object detection model. It introduces a more lightweight and powerful feature extraction backbone, Efficientnetv2, and utilizes the He-Kaiming weight initialization method to extract strong multi-scale feature maps. In training, a more efficient dynamic contrastive denoising training method is employed. This approach makes the model lightweight and accurate for 3C detection. This method outperforms leading detection algorithms in both accuracy of experimental results and parameter efficiency.

Keywords: 3C industry, object detection, DETR decomposition

1 Introduction

Intelligent manufacturing (IM) epitomizes the deep integration of information technology and industrialization [1]. In the 3C industry, where manual labor prevails due to high repeatability in assembly processes and rapid product updates, the imperative to transition from manual and semi-automatic methods to intelligent automation is increasingly pressing, especially with China's aging population. However, traditional visual industrial robots face limitations in handling the industry's characteristics of numerous irregular small parts, semi-flexible targets, and complex assembly processes. The incorporation of machine vision technology into industrial assembly robots has emerged as a pivotal direction in intelligent manufacturing, addressing these challenges [2]. By significantly enhancing the visual perception capability of 3C industrial robots, intelligent visual recognition technology enables them to autonomously identify assembly parts' category and position information and handle complex assembly situations independently. This integration elevates the level of intelligence in assembly robots, enhances the flexibility and robustness of industrial production lines, and further improves production efficiency. Despite the current challenge of large-sized smartphone component detection algorithm models consuming significant computational resources, this approach aligns with the demand for precision assembly, rapid iteration, and small-batch customization in the 3C industry, fostering a seamless transition towards intelligent automation.

The rapid development of deep learning has made CNN-based object detection algorithms the mainstream approach.Object detection algorithms based on CNN can be classified into single-stage, two-stage, and multi-stage methods, depending on the detection process stages. R-CNN (Regions with Convolutional Neural Network), proposed by Girshick et al. [3], is a two-stage object detection algorithm that first generates candidate regions in the image, then classifies and regresses these regions. By introducing deep learning methods into the traditional approach, R-CNN greatly improves the detection accuracy. Cascade-RCNN, proposed by Cai et al. [4], is a multi-stage object detection algorithm. It follows a similar detection process as the two-stage approach but differs in that it iteratively refines the candidate regions, leading to improved detection accuracy without sacrificing detection speed.

Although two-stage object detection algorithms offer high accuracy, their large size hinders real-time detection. As a result, one-stage end-to-end algorithms such as YOLO [5-7], RetinaNet [8], and EfficientDet [9] have been proposed, which excel in real-time object detection.

Both two-stage and single-stage end-to-end object detection algorithms are based on Convolutional Neural Networks (CNNs). CNNs primarily utilize local processing, which means they often struggle to effectively

* Corresponding Author

capture the deep interconnections between global features of an image. This limitation can lead to performance bottlenecks in complex image recognition tasks. In contrast, the Transformer model incorporates an architecture based on attention mechanisms, which assesses and utilizes the relationships between data elements, facilitating comprehensive global information processing. This capability allows the Transformer to not only identify local features but also capture connections across the entire image, significantly enhancing the model's predictive accuracy and robustness. Therefore, integrating the advantages of the Transformer into object detection algorithms can not only compensate for the deficiencies of traditional CNNs in processing global features but also substantially improve the overall performance of object detection, especially in highly complex and dynamically changing visual environments.

DETR (DEtection TRansformer) [10] is the inaugural end-to-end detection model using the Transformer architecture, framing object detection as direct set prediction and separating predictions from specific target positions. It integrates CNN and Transformer structures for object detection. Zhu et al. [11] identified the attention mechanism of the Transformer as the primary cause for the slow convergence of DETR, attributing it to its modeling of dense relationships between global features.Consequently, it takes a long time for the model to focus on meaningful sparse positions. To address this, they introduced Deformable DETR, which leverages the idea of deformable convolution [12]. By performing sparse sampling on different hierarchical feature maps, the model can prioritize learning meaningful key positions and accelerate convergence speed. Additionally,Deformable DETR enhances small object detection accuracy by using attention mechanisms to aggregate information across multi-scale feature maps.In the same year, Sun et al. [13] introduced the encoder-only version of DETR as a means to address the issue of slow convergence resulting from the cross-attention module in the decoder of DETR. Expanding upon this concept, they also proposed two ensemble prediction models known as TSP-FCOS and TSP-RCNN. These models devised new bipartite matching strategies to address the instability caused by the Hungarian loss in the original DETR, achieving faster convergence during ensemble prediction training.

This paper delves into the challenge of detecting smartphone components in the 3C industry, particularly in environments characterized by flexible printed circuits and small, diverse parts. Given the constraints of computational resources in industrial settings, there is a critical need to improve object feature learning and detection performance. To effectively address the challenges associated with detecting smartphone component assembly in the 3C industry, we have developed a novel, lightweight network architecture that significantly enhances detection accuracy while maintaining efficiency. Central to our model is the integration of EfficientNetV2 [14] as the backbone, which is renowned for its compact structure and superior feature extraction capabilities compared to traditional models. This backbone facilitates the extraction of robust feature maps, which are further optimized through the implementation of the He-Kaiming weight initialization method, renowned for its ability to maintain a balanced variance in activations, thus preventing the vanishing gradient problem during deep network training. Moreover, we employ a dynamic contrastive denoising training approach that not only reduces noise in the training data but also adapts the learning process based on the current state of the model, thereby enhancing both the lightness and the precision of the model. This approach ensures that our network remains not only agile in processing speeds but also exceptionally accurate in the detection of intricate components in the fast-paced 3C industry.

2 Related Work

2.1 DETR and Its Variants

DETR simplifies object detection by eliminating components such as NMS and anchor generation. Based on its excellent performance, many variant algorithms based on it have been produced. Meng et al. [15] introduced Conditional DETR as a solution to the primary challenge of slow convergence during DETR training. It learns conditional spatial queries from decoder embeddings for multi-head cross-attention, narrowing the spatial range for object classification and box regression. This reduces reliance on content embeddings and simplifies training. Wang et al. [16] presented Anchor DETR, where the query design of this study incorporates the use of anchor points, a prevalent approach in CNN-based detectors. Consequently, each object query is directed towards the target in close proximity to its corresponding anchor. Dai et al. [17] proposed Dynamic DETR, which integrates dynamic attention in both encoder and decoder stages of DETR to address its limitations of small feature resolution and slow training convergence. Additionally, a dynamic decoder replaces the cross-attention module with ROI-based dynamic attention in the Transformer decoder, significantly easing learning and speeding up conver-

gence. DAB-DETR [18] introduces Dynamic Anchor Boxes (DAB) as the definition for DETR queries, effectively bridging the divide between conventional anchor-based detectors and class-agnostic detectors. DN-DETR [19] addresses slow convergence by tackling the instability of bipartite matching, which causes inconsistent optimization objectives early in training. The approach introduces noisy GT boxes to the Transformer decoder and trains the model to reconstruct original boxes, simplifying bipartite matching and speeding up convergence. DINO [20] enhances the performance and efficiency of prior class-agnostic DETR models as an advanced end-to-end object detector. It incorporates contrastive denoising training, mixed query selection with anchor initialization, and a double look-forward scheme for box prediction. Co-DETR [21] proposes a novel collaborative mixed-task training scheme to learn more efficient DETR-based detectors from multiple label assignment strategies.

DETR models, despite their innovation, have been criticized for not effectively addressing the high computational costs associated with their operation, which curtails their practical utility and undermines their capability to exploit the full advantages of post-processing-free operations such as non-maximum suppression (NMS). The RT-DETR model [22] addresses this limitation by introducing a hybrid encoder that optimizes the processing of multi-scale features through a sophisticated mechanism of decoupled intra-scale interactions coupled with efficient cross-scale fusion. This enhancement significantly reduces inference delays typically associated with NMS, thereby facilitating real-time object detection. Furthermore, in comparative evaluations on the COCO dataset, RT-DETR has demonstrated superior performance in both speed and accuracy, outpacing all same-scale YOLO [5] detectors. Consequently, within the domain of 3C object detection, RT-DETR presents substantial advantages, suggesting a promising direction for future research and application in environments demanding high efficiency and precision.

2.2 Lightweight Neural Network

Recent research has focused on the development of compact and efficient neural networks specifically designed for resource-limited industrial environments. These networks aim to minimize computational load and parameter count without compromising the integrity of model performance. The primary goal of designing lightweight networks is to refine computation methods, particularly convolution techniques, to significantly reduce the number of network parameters while maintaining robust network representation capabilities. This approach not only optimizes performance in constrained settings but also enhances the efficiency of data processing, thereby facilitating more advanced applications in real-time industrial operations.Some research methods include dilated convolution, deformable convolution, and depthwise separable convolution. For example, SqueezeNet [23] achieves parameter reduction by compressing the channel count within feature maps through the utilization of 1x1 convolution kernels. Additionally, the fire module combines dilated convolution and 1x1 convolution, further enhancing feature extraction efficiency. The MobileNet [24] series, developed by Google's research team, comprises lightweight convolutional neural network architectures designed to markedly reduce network parameters and computational burden while preserving high performance. They are particularly su for resource-constrained environments like mobile devices and embedded systems. ShuffleNet's [25] brings a significant innovation to the field of neural network architectures through the introduction of group convolution and channel shuffling mechanisms. The channel shuffling mechanism, in particular, plays a crucial role by facilitating effective information exchange across channels. This exchange enhances the interaction between feature maps, thereby significantly improving the overall performance of the network. Such advancements make ShuffleNet particularly suitable for environments where computational resources are limited, such as in mobile devices and embedded systems. This architecture not only optimizes the computational efficiency but also maintains competitive accuracy, underscoring its utility in advancing the capabilities of lightweight neural networks.

2.3 Parameter Initialization Method

Convolutional neural networks (CNNs) have dramatically transformed a variety of visual tasks, yet the process of training these models from scratch presents considerable challenges for researchers in the field. To circumvent these difficulties, the prevailing strategy involves the utilization of larger, pre-trained models which are then fine-tuned or specifically adapted to distinct visual tasks. This approach primarily stems from the complex and delicate nature of network initialization, where even minor inaccuracies in setting the initial weights can lead to significant issues, such as the phenomena of gradient vanishing or explosion. These issues, in turn, result in suboptimal convergence rates during the training phase. This widespread reliance on pre-trained models not only

highlights the challenges associated with the effective initialization of CNNs but also underscores the importance of these foundational models in achieving optimal performance across a spectrum of visual tasks. This strategic emphasis on pre-trained models thereby plays a pivotal role in simplifying the training process and enhancing the overall efficacy and efficiency of CNN deployment in diverse applications.

Philipp et al. [26] introduced a swift and straightforward data-dependent initialization process that establishes the network weights in a manner that promotes balanced training rates across all units, mitigating the issues of gradient vanishing or explosion. A novel neural network architecture has emerged, known as networks with block-sparse weights. These kernels enable the efficient evaluation and differentiation of linear layers, including convolutional layers, while also offering the flexibility to configure block-sparse patterns within weight matrices. And a class of weight initialization conditions called random orthogonal initialization, which, like unsupervised pre-training, has deep independent learning time.

Xavier initialization, alternatively termed Glorot initialization, was originally proposed by Glorot et al. [27] as a pivotal strategy to mitigate the limitations associated with random initialization in neural network training. The crux of their proposal lies in aligning the distributions of inputs and outputs to maintain consistency across layers, thereby averting the propensity for subsequent layers' activation function outputs to gravitate towards zero. By ensuring that the variance of the activations remains relatively consistent across layers, Xavier initialization fosters more stable and efficient training dynamics, ultimately enhancing the convergence properties of the neural network. This initialization technique has since become a cornerstone in the design and training of deep learning models, underscoring its significance in facilitating robust and expedited convergence during the optimization process.

In contrast to Xavier initialization, Kaiming initialization [28] does not require each layer's output mean to be 0, nor does it require f'(0) = 1. In Kaiming initialization, distinct initialization strategies are employed for each forward and backward pass. The objective is to maintain a variance of 1 for both the output of each layer during forward propagation and the gradients during backward propagation.

3 Model

3.1 Preliminaries

DETR pioneered the application of the Transformer model to object detection tasks.The image first goes through a traditional CNN to extract features, then the output of the CNN is directly fed into the Transformer network. The Transformer outputs a set of predictions, each including the bounding box's center coordinates, width, height, and class. A bipartite graph matching strategy is employed to associate the predicted boxes with the ground truth (GT) boxes in order to compute the loss.

In the DETR model, the cross-attention mechanism necessitates a simultaneous match between the query's content embedding and both the content and spatial embeddings in the key, which imposes stringent quality requirements on the content embeddings. Observations from the training process, particularly after 50 epochs, indicate that DETR's performance is hampered by suboptimal content embedding quality, leading to an inability to accurately narrow down the object search scope, and consequently, slow convergence rates. This sluggish convergence is primarily due to DETR's heavy reliance on high-quality content embeddings for pinpointing the extremity regions of objects, which are pivotal for accurate object localization and recognition.To address the challenges associated with the dependency on high-quality content embeddings, the Conditional DETR model introduces a decoupling mechanism in the decoder's cross-attention functionality. In this architecture, content embeddings are tasked exclusively with identifying regions pertinent to the object's appearance, eliminating the need for alignment with positional embeddings. This separation allows the conditional positional embeddings to explicitly target and refine the search towards the object's extremity regions, thus enhancing the efficiency and precision of the object localization process.

In the Conditional DETR paradigm, the construction of each query is orchestrated by the fusion of a content query (cq) and a spatial query (pq), thereby orchestrating the synthesis of a prospective detection outcome at the terminal phase of each decoder layer. This architectural configuration engenders the emergence of a candidate detection outcome, which is pivotal in discerning the salient features within the visual input. The crux of the attention mechanism lies in the computation of attention weights, a process facilitated by the dot product interaction between the queries and the keys. This interactional modality assumes paramount importance in delineating the relative significance of different elements within the input sequence, thereby underpinning the discernment of

pertinent visual cues essential for accurate object detection.

$$\begin{aligned}&(c_q + p_q)^T (\mathrm{c}_k + p_k)\\&= c_q^T ck + c_q^T p_k + p_q^T c_k + p_q^T p_k \ .\\&= c_q^T c_k + c_q^T p_k + o_q^T c_k + o_q^T p_k\end{aligned} \tag{1}$$

In DAB-DETR, the object query is explicitly represented as coordinates [x, y, w, h]. Within the framework of Conditional DETR, the process involves transforming decoder embeddings through a Feedforward Neural Network (FFN) to derive offsets. These offsets, when amalgamated with reference points, result in a refined positional encoding. This refined encoding is subsequently integrated into the Cross Attention mechanism of the Decoder. Distinctively, DAB-DETR performs an iterative update of the query at the conclusion of each decoder layer. This update process, mirroring the Conditional DETR approach, leverages an FFN to generate offsets from the decoder embeddings. The resultant offsets are combined with reference points to achieve a multi-tiered refinement of the positional encoding. In this architecture, each layer's computation of relative offsets is guided by supervision from the ground truth, with the optimization confined to the parameters specific to the current layer.

Drawing from Conditional DETR, DAB-DETR introduces a novel cross-attention mechanism that integrates positional and content information as both queries and keys. This methodology enables the systematic decoupling of the influences of content and position on query-feature similarity into a refined dot product calculation between queries and keys. To advance the sophistication of positional embeddings, DAB-DETR incorporates conditional spatial queries. More precisely, it utilizes a Multilayer Perceptron (MLP), MLP(csq): $\mathbb{R}$^D → $\mathbb{R}$^D, to generate a scale vector predicated on content information. This scale vector is subsequently utilized in an element-wise multiplication with the positional embeddings, thereby achieving a targeted refinement of these embeddings within the model's architecture.

$$\begin{aligned}&Cross - Attn : Q = Cat(C_q, PE(x_q, y_q) \bullet MLP^{(csq)}(C_q))\\&K_{x,y} = Cat(F_{x,y}, PE(x, y)), V_{x,y} = F_{x,y} \ .\end{aligned} \tag{2}$$

where $F_{x,y} \in \mathbb{R}^D$ represents the image feature located at position (x, y), and the symbol $\bullet$ signifies element-wise multiplicatio.

DN-DETR targets the issue of slow convergence inherent in DETR by analyzing the discrete and stochastic nature of the match between Ground Truth (GT) boxes and predicted boxes, a process typically governed by the Hungarian algorithm. This mechanism results in a dynamic and unstable matching environment, where each query's predicted detection box may correspond to different GT boxes, thus complicating the offset learning process. The core of the instability lies in the Decoder's task of learning offsets relative to an anchor, with the variability in GT box matching leading to erratic learning of offsets. To mitigate these challenges, DN-DETR innovatively employs a denoising task as a strategic shortcut, enabling the direct learning of relative offsets and effectively sidestepping the complex matching process. As a result, DN-DETR simultaneously trains on two distinct tasks: the conventional matching task and the supplementary denoising task, thereby refining the learning pathway and enhancing stability in offset learning.

DINO, a DETR-like model, features a backbone network, multi-layer Transformer encoder and decoder, and multiple prediction heads. According to DAB-DETR [18], the model assigns queries in the decoder as dynamic anchor boxes. Similarly, DN-DETR [19] introduces ground truth labels and noisy boxes to the Transformer decoder layer, enhancing bipartite matching stability during training.To enhance computational efficiency, we also employ deformable attention [11]:

$$DeformAttn(z_q, p_q, x) = \sum_{m=1}^{M} W_m [\sum_{k=1}^{K} A_{mqk} \cdot W_m' x(p_q + \Delta p_{mqk})] \ . \tag{3}$$

The model employs a contrastive denoising training approach to enhance one-to-one matching by adding positive and negative samples with identical ground truths. As depicted in Fig. 1, two different noises are applied to the same truth, one labeled as positive and the other as negative, which prevents repetitive detections of the same object. Additionally, the model uses a hybrid query selection method for better query initialization, selecting

initial anchor boxes from the encoder's outputs as positional queries. Furthermore, it introduces a 'look forward twice' scheme that utilizes gradients from later layers to refine parameters in earlier layers, optimizing performance.

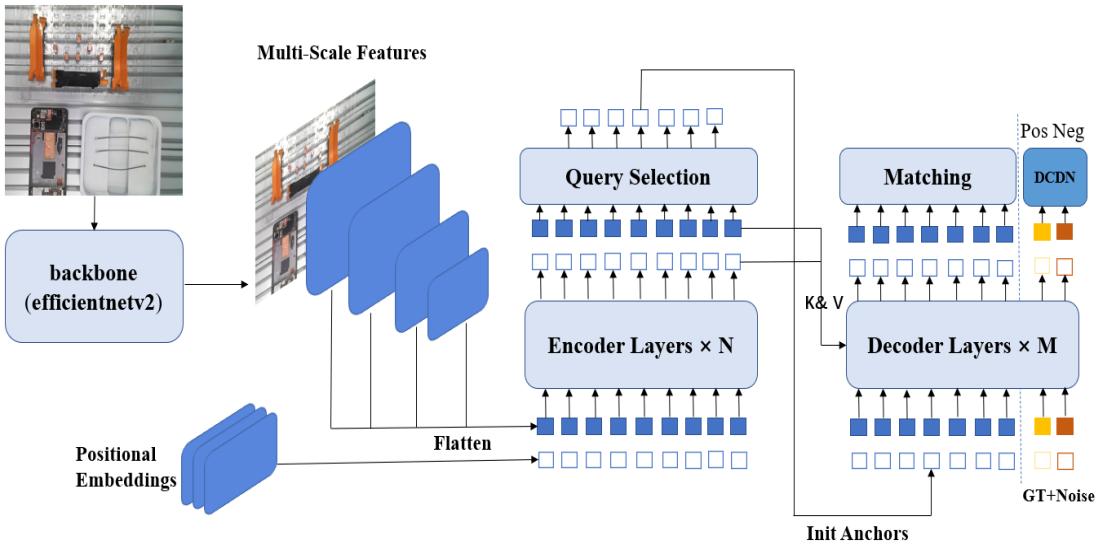


Fig. 1. The framework of DINO model

(Our decoder also includes a dynamic contrast denoising (DCDN) section with positive and negative samples.)

The study of Conditional DETR [15] and DAB-DETR [18] reveals that DETR queries consist of positional and content components, referred to as position queries and content queries in this paper. DAB-DETR [18] defines each position query as a 4D anchor box (x, y, w, h), where x and y denote the center coordinates, and w and h the width and height. This definition allows for dynamic optimization of anchor boxes across decoder layers.

3.2 Efficientnetv2

EfficientNetV2 [14], an innovative convolutional neural network architecture devised by Google in 2021, marks a significant advancement in the lineage of EfficientNet models. Renowned for its superior accuracy coupled with a reduced parameter count, EfficientNetV2 offers expedited inference speeds, surpassing its predecessors in efficiency and performance. The architectural enhancements introduced in EfficientNetV2 optimize the utilization of parameters, enabling the network to achieve remarkable accuracy levels while demanding fewer computational resources during both training and inference phases. This evolution represents a pivotal stride towards addressing the ever-growing demands for efficient yet high-performing deep learning models across various applications and domains.

The model integrates training-aware neural architecture search with model scaling, enhancing both training speed and parameter efficiency. EfficientNet's main training bottleneck is the extensive use of depthwise convolutions, which have fewer parameters and FLOPs than standard convolutions. Fused-MBConv, introduced by Gupta & Tan [29], combines the depthwise 3x3 and pointwise 1x1 convolutions in MBConv [30] into a single regular 3x3 convolution, as depicted in the figure. This model uses neural architecture search (NAS) to optimize the combination of MBConv and Fused-MBConv modules.

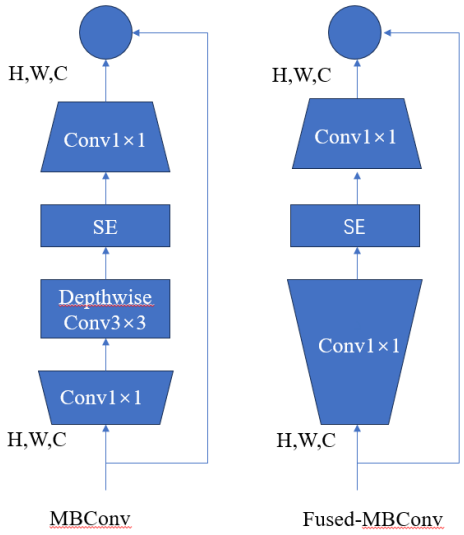


Fig. 2. Structure of MBConv and Fused-MBConv

EfficientNetV2 heavily utilizes MBConv and Fused-MBConv, as shown in Fig. 2. It spans from Stage0 to Stage7. The initial stage (Stage0) features a convolutional layer with a kernel size of 3 and a stride of 2. Subsequent stages, Stage1 to Stage3 and Stage4 to Stage6, incorporate repeated stacking of Fused-MBConv and MBConv structures, while Stage7 employs a standard 1x1 convolutional layer. EfficientNetV2 uses smaller 3x3 convolutional kernels but compensates for the reduced receptive field caused by these smaller kernels by adding more layers. This approach allows information from larger regions of the input image to be incorporated into the final output, thereby maintaining or enhancing network accuracy with smaller convolutional kernels. The structure of EfficientNetV2-S is depicted in Table 1.

Table 1. The architecture of EfficientNetV2-S, consisting of MBConv and Fused-MBConv blocks, is depicted in Fig. 2

Stage	Operator	Stride	#Channels	#Layers
0	Conv3*3	2	24	1
1	Fused-MBConv1, k3*3	1	24	2
2	Fused-MBConv4, k3*3	2	48	4
3	Fused-MBConv4, k3*3	2	64	4
4	MBConv4, k3*3, SE0.25	2	128	6
5	MBConv6, 3*3, SE0.25	1	160	9
6	MBConv6, 3*3, SE0.25	2	256	15
7	Conv1, k3*3	-	1280	1

In our object detection framework, we leverage the potent feature extraction capabilities of EfficientNetV2-S and ResNet50, which are pretrained on the extensive ImageNet-22k dataset [31], followed by fine-tuning on our specific dataset. This amalgamation empowers our model with a robust foundation for discerning intricate visual features across a plethora of object categories. A pivotal aspect enhancing our model's efficacy is the incorporation of a deformable attention module, which seamlessly integrates with the multi-scale feature maps obtained during the feature extraction process. This strategic integration not only facilitates the aggregation of contextual information across varying spatial scales but also fosters the discernment of nuanced spatial dependencies crucial for accurate object localization and classification. By harnessing the deformable attention mechanism, our model adeptly captures subtle spatial nuances inherent in diverse object scales, thereby bolstering its object detection performance comprehensively. This amalgamation of state-of-the-art backbones with deformable attention mechanisms underscores our commitment to advancing the frontiers of object detection by amalgamating cutting-edge methodologies with established architectural prowess.

3.3 HKM Parameter Initialization

Weight initialization involves setting initial network parameters (weights and biases) to prevent issues like vanishing or exploding gradients, thereby accelerating network convergence and enhancing model performance and accuracy. The Xavier initialization method performs poorly with ReLU layers, mainly because ReLU maps negative values to zero, affecting the overall variance. Additionally, the Xavier initialization method is limited to certain types of activation functions: those that require symmetry around zero and linearity. ReLU activation function does not meet these requirements, and experiments have shown that Xavier initialization is indeed not suitable for ReLU activation.

To address these limitations, improvements have been made by Kaiming initialization proposed by He et al. [30] Initially, Kaiming initialization was mainly applied to computer vision and convolutional networks. Glorot and Bengio [26] introduced a method called "Xavier" initialization, this method assumes linearity in the activation function, which is ineffective for ReLU and PReLU activation functions. To overcome these limitations, this paragraph introduces a theoretically more reliable initialization method that takes into account ReLU and PReLU. This new method can successfully converge very deep models with up to 30 convolutional or fully connected layers, while the "Xavier" method.

Forward Propagation Case. For the convolution layer, the output response is

$$\mathrm{y}_1 = W_l x_l + b_l . \tag{4}$$

We make the initialized elements independent of each other and assume that the elements in x_l also share the same distribution, while x_l and W_l are independent of each other. Then we have:

$$Var[y_l] = n_l Var[w_l x_l] . \tag{5}$$

After assuming a zero mean for w_l, the variance of the product of independent variables is obtained as follows:

$$Var[y_l] = n_l Var[w_l] E[x_l^2] . \tag{6}$$

Here $E[x_l^2]$ is the expectation of x^2. And, it is worth noting $E[x_l^2] \neq Var[\mathrm{x}_l]$, unless the mean of x_l is zero, for ReLU activation, $\mathrm{x}_l = \max(0, y_{l-1})$, so its average is not zero.

If we assume w_{l-1} has a symmetric distribution near 0 and $b_{l-1} = 0$, the mean of theta is zero and there is also a symmetric distribution near 0. This results $Var[x_l^2] = {}^1/_2\, Var[y_{l-1}]$ when f is ReLU. Bring it into the above equation:

$$Var[y_l] = \frac{1}{2} n_l Var[w_l] Var[y_{l-1}] . \tag{7}$$

With L layers assembled, we have:

$$Var[y_l] = Var[y_1]\left(\prod_{l=2}^{L} \frac{1}{2} n_l Var[w_l]\right) . \tag{8}$$

Proper initialization methods should steer clear of exponentially diminishing or amplifying the amplitude of the input signal. Consequently, we anticipate the aforementioned product to adopt an inherent scalar value (e.g., 1), under specific conditions:

$$\frac{1}{2} n_l Var[w_l] = 1 . \tag{9}$$

This leads to a zero-mean Gaussian distribution having a standard deviation (std) of $\sqrt{\frac{2}{n_l}}$. Moreover, we set b = 0.

Backward Propagation Case. For backpropagation, the gradient of the transition layer is:

$$\Delta x_l = \widehat{W}_l \Delta y_l \,. \tag{10}$$

Here we use Δx and Δy to represent the gradient ($\frac{\partial \varepsilon}{\partial x}$ and $\frac{\partial \varepsilon}{\partial y}$). Δy represents k-by-k pixels in d channels, and is reshaped into a k2d-by-1 vector. We denote $\hat{n} = k^2 d$ Take note that $\hat{n} \neq n = k^2 c$. $\widehat{W}$ is a c-by- $\hat{n}$ matrix where thefilters are rearranged in the manner of back-propagation.

In backpropagation, we also have $\Delta y_l = f'(y_l)\Delta x_{l+1}$. where f' represents the derivative of f. In the case of ReLU, f' is either 0 or 1, with equal probabilities. Let's assume that $f'(y_l)$ and Δx_{l+1} are independent. Thus we have $E[\Delta y_l] = E[\Delta x_{l+1}]/\ 2 = 0$, and also $E[(\Delta y_l)^2] = Var[\Delta y_l] = {}^1\!/_2\ Var[\Delta x_{l+1}]$, then, calculate the variance of the gradient.

$$Var[\Delta x_l] = \hat{n}_l Var[wl] Var[\Delta yl] \ . \tag{11}$$

$$Var[\Delta x_l] = \frac{1}{2} \hat{n}_l Var[w_l] Var[\Delta x_{l+1}] \,. \tag{12}$$

Putting the L layers together, we have:

$$Var[\Delta x_2] = Var[\Delta L + 1](\prod_{l=2}^{L} \frac{1}{2} \hat{n}_l Var[w_l]) \,. \tag{13}$$

We consider that the gradient is not a sufficient condition for being exponentially large/small:

$$\frac{1}{2} \hat{n}_l Var[w_l] = 1 \,. \tag{14}$$

For the first layer (L= 1), computation of Δx1 is unnecessary as it pertains to the image domain. However, Eqn.(14) can still be applied to the first layer, similar to forward propagation, since single-layer factors do not exponentially increase the overall product. If initialization scales the backward signal suitably, the forward signal is similarly affected, and vice versa. In this study, both methods converge across models.

Efficientnetv2 utilizes the SILU [32] activation function, an enhancement over the RELU type. Both functions are depicted in Fig. 3 below, demonstrating compatibility with He Kaiming's initial method.

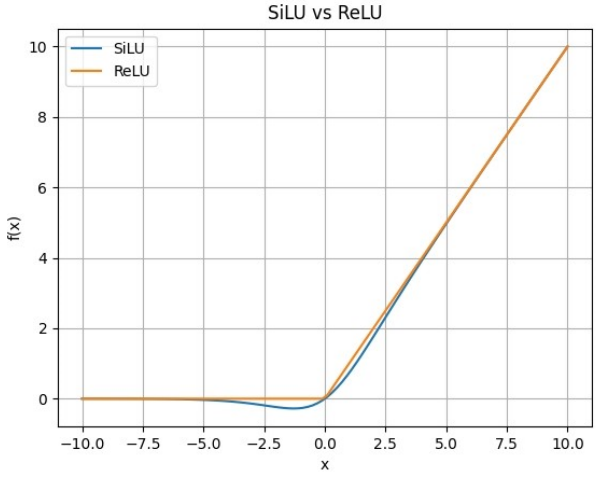


Fig. 3. SiLU and ReLU Activation function

3.4 Dynamic Contrastive Denoising Training

Dynamic Contrastive Denoising Training (DCDN) emerges as a robust methodology in enhancing model accuracy within computer vision domains. It amalgamates the efficacy of contrastive learning and denoising autoencoders to refine feature representations essential for various tasks. Contrastive learning, primarily employed in unsupervised or semi-supervised paradigms, fosters feature enhancement by orchestrating a process of pulling similar data points together while concurrently pushing dissimilar ones apart. This approach capitalizes on the fundamental principle of maximizing the similarity between positive pairs and minimizing it between negative pairs, thereby encouraging the model to discern subtle differences crucial for accurate classification or representation learning. On the other front, denoising autoencoders, rooted in unsupervised learning frameworks, contribute significantly by inducing noise into input data and training the model to reconstruct the original, unadulterated data. Through this process, the autoencoder learns to denoise and distill the salient features from the noisy input, consequently facilitating robust feature extraction and representation learning. By integrating these two methodologies synergistically, DCDN not only leverages the discriminative power of contrastive learning but also harnesses the denoising capabilities of autoencoders, thereby yielding superior performance in various computer vision tasks. The fusion of these techniques not only enhances model robustness but also fosters a deeper understanding of the underlying data manifold, which is pivotal for advancing the state-of-the-art in computer vision research.

DCDN combines these two methods. During the training process, the noise level and contrastive strength are dynamically adjusted based on the requirements of the task. This approach enables the model to concentrate on varying feature representations at distinct stages, effectively capturing the data's hierarchical structure and enhancing the accuracy of computer vision tasks to a certain degree. DN-DETR features a hyperparameter λ to regulate noise scale, ensuring that generated noise does not exceed λ, as it reconstructs the ground truth (GT) from moderately noisy queries. In the DINO method, there are two hyperparameters λ1 and λ2, the scales of λ1 and λ2 are shown in Fig. 4, where λ1 < λ2. As demonstrated in Fig. 5, positive queries within the inner square possess a noise scale smaller than λ1 and aim to reconstruct their corresponding GT boxes. Negative queries between the inner and outer squares have a noise scale greater than λ1 and smaller than λ2. They are anticipated to predict "no object". Typically, a smaller λ2 is chosen since hard negative samples are closer.

GT boxes enhance performance, with each CDN group holding a set of positive and negative queries. For an image with n GT boxes, a CDN group contains 2×n queries, each GT box producing one positive and one negative query. Like DN-DETR, multiple CDN groups are used to improve our method's effectiveness. The reconstruction losses are l1 and GIOU for box regression, and focal loss [19] for classification.

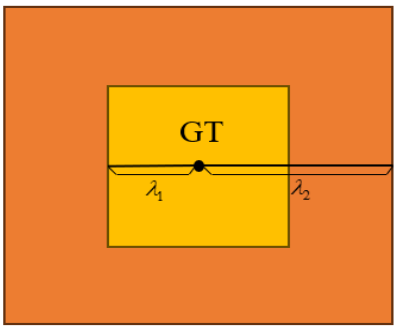


Fig. 4. λ1 and λ2 in GT box

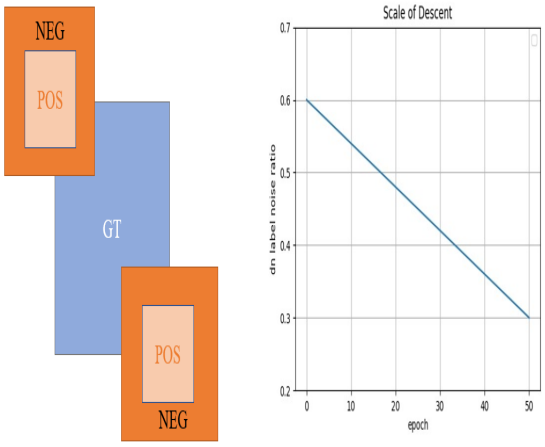


Fig. 5. Assuming that the center of the square is the GT box, the points inside the inner square are regarded as positive examples, and the points between the inner square and the outer square are regarded as negative examples

In this paper, the dynamic contrastive denoising training method is utilized, deriving its efficacy from the capacity to attenuate ambiguity and identify superior anchors (queries) for bounding box prediction. Ambiguity arises when multiple anchors are proximate to an object, posing challenges for the model in anchor selection. To better capture the hierarchical structure of the data by focusing on different feature representations at different stages, the noise level is dynamically adjusted and decreases during the training process. The equation for the decreasing noise level is: $DN\_LNR = -0.006EPOCH + 0.6$

DN_LNR (dn_label_noise_ratio) represents the probability of randomly flipping noisy labels in the training data. We achieve dynamic contrastive denoising training by continuously reducing this probability during the training process. The decreasing scale of dynamic contrast denoising in the training process is shown in Fig. 6.

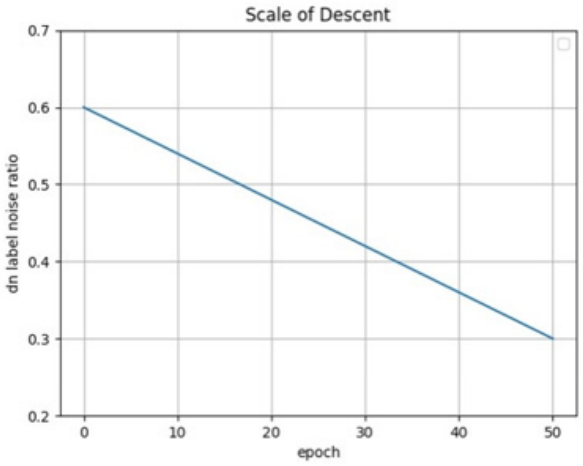


Fig. 6. The decreasing scale of dynamic contrast denoising in the training process

Specifically, dynamic contrastive denoising training (DCDN) is implemented by introducing positive and negative samples around the true labels of targets. Positive samples are very close to the actual targets but with slight variations or noise, whereas negative samples are significantly different from the actual targets in the feature space. During the training, the model learns to differentiate positive from negative samples, thereby enhancing its ability to detect subtle feature differences. This method is particularly suitable for detecting small objects in target detection tasks or those that are difficult to distinguish in complex backgrounds. The key to the DCDN strategy is how to dynamically adjust the noise level and contrast intensity. In the early stage of training, a higher noise level and contrast intensity enable the model to swiftly capture the rough features of the target. As the training progresses, the noise level and contrast intensity are gradually reduced, allowing the model to focus more on learning the detailed features of the target. This gradual refinement process helps the model continuously improve its detection performance throughout the training cycle.

Through this dynamic adjustment mechanism, the DCDN training strategy can effectively balance the model's rapid learning in the initial stage and fine-tuning in the later stage, ultimately achieving high-precision detection of targets, especially for small-sized targets or those in complex backgrounds. This training strategy not only improves the model's performance but also enhances its adaptability to 3C scenarios.

The DCDN strategy also optimizes the loss function to moderate the effects of various error types on model performance. By tuning the weights within this function, the model effectively balances reducing false positives (misidentifying the background as the target) and false negatives (overlooking the actual target), enhancing detection accuracy. Additionally, we adjust the cost coefficients for classification loss and bounding box loss to further refine DCDN's adaptability.

4 Experimental Results

4.1 Datasets

The Microsoft Azure Kinect sensor, attached to the manipulator's end and acting as its 'eyes,' is employed in dataset generation. It simulates diverse scenarios across various times and lighting conditions to broaden the dataset's diversity and enhance the model's generalization capabilities. In the context of the 3C scene, which features semi-flexibility and small-scale objects, a specific strategy for enhancing small objects is implemented in the dataset construction. This strategy focuses on incorporating numerous small targets and ensuring high-resolution imagery of these elements within the scenes, which helps the network model to adequately learn data features pertinent to small objects. The dataset comprises 2000 images of five components: Flexible Printed Circuit (FPC), Coaxial Cable (COAX), SIM Card Slot, Front Camera (CAM), and Mobile Phone (MP) models, each with a resolution of 2048×1536. We trained our model using this self-built dataset, dividing it into training, validation, and test sets at an 8:1:1 ratio for our experiments.

4.2 Implementation and Experimental Setup

In this experiment, we utilized a server equipped with an Intel(R) Xeon(R) Silver 4110 CPU, 64GB RAM, Ubuntu OS, and 8 RTX2080Ti GPUs. We employed the PyTorch framework, training a model comprising 6-layer Transformer encoders and decoders, each with a hidden feature dimension of 256, at a batch size of 2. Performance testing of DINO was limited to 4-scale mode, alongside comparative and ablation studies, detailed in Table 2.

Table 2. Experimental parameter setting

lr	0.0001	num queries	900
lr_backbone	1e-5	enc_n_points	4
enc_layers	6	dec_n_points	4
dec_layers	6	set cost class	2.0
dim feedforward	2048	set cost bbox	5.0
hidden dim	256	set cost giou	2.0
dropout	0.0	focal alpha	0.25
nheads	8	batch	2

4.3 Main Results

In the comparative evaluation of object detection methodologies, our study juxtaposed the performance of our proposed DINO (OURS) model against established benchmarks including Faster RCNN, Mask RCNN, SSD, RetinaNet, DETR, DAB-DETR, and DN-DETR. Table 3 delineates the results, where the DINO (OURS) model distinguishes itself by achieving a commendable Average Precision (AP) of 60.2 following 50 epochs. Remarkably, it exhibits an AP50 of 94.5 and an AP75 of 65.6, surpassing the performance metrics of all scrutinized models. Conversely, the traditional Faster-RCNN model, while boasting a notable AP50 of 95.0, only records an overall AP of 57.6, underscoring its limitations particularly in the realm of small object detection (APS) where it demonstrates a modest 14.0 AP. This nuanced analysis highlights the efficacy of the DINO (OURS) model in addressing challenges associated with granular object detection tasks.

The DAB-DETR and DN-DETR models, conceived as refinements to ameliorate the DETR's protracted convergence, have demonstrated incremental advancements with recorded average precisions (APs) of 56.9 and 56.5, respectively. These iterations exhibit heightened proficiency in detecting medium and large-scale objects, indicative of their capacity for robust detection within such contexts. However, despite these enhancements, the persistent challenge lies in the detection of diminutive objects, where performance remains circumscribed. Noteworthy is the DETR (DC5) model's underwhelming AP of 46.3, signaling its deficient detection capabilities particularly in intricate environmental settings. Hence, while strides have been made to enhance overall detection efficacy, the pursuit of comprehensive object detection, especially in scenarios with intricate spatial configurations, remains a focal point for further refinement and innovation within the realm of transformer-based models.

Table 3. Results for DINO and other detection models

Model	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params
Faster-RCNN	50	57.6	**95.0**	**68.2**	14.0	48.7	69.4	40M
Mask-RCNN	50	55.6	92.8	58.0	17.7	47.3	61.5	54M
SSD	50	54.6	89.5	56.7	5.4	33.8	66.2	41M
Retinanet	50	56.9	90.5	62.5	11.5	**49.2**	65.4	32M
DETR(DC5)	50	46.3	83.6	49.7	2.4	22.4	54.5	41M
DAB-DETR	50	56.9	84.1	58.6	12.3	40.1	66.6	44M
DN-DETR	50	56.5	82.7	59.5	11.7	36.3	65.0	48M
DINO	50	58.1	93.6	63.4	17.9	45.6	67.7	47M
DINO (Ours)	50	**60.2**	94.5	65.6	**19.3**	46.7	**70.3**	**34M**

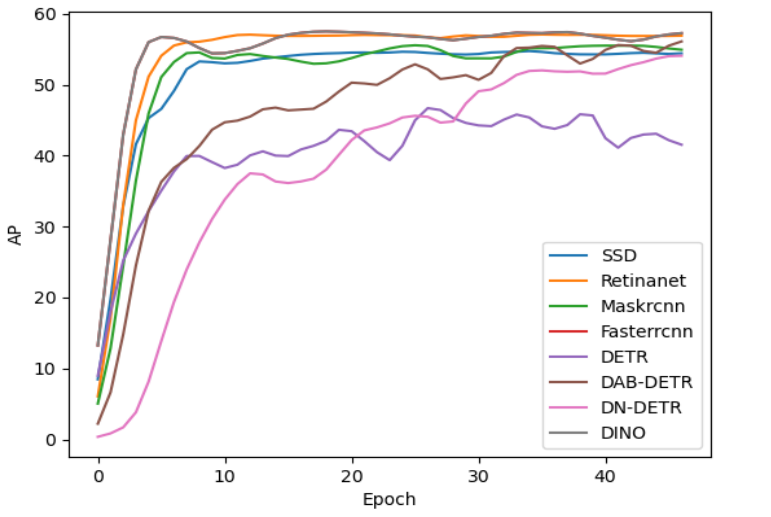


Fig. 7. Algorithm accuracy curve

The empirical evidence from the provided data unequivocally underscores the superior performance of the DINO (OURS) model in object detection tasks, particularly in the nuanced realm of detecting objects of varying sizes. Notably, its adeptness in discerning small objects surpasses that of its counterparts by a significant margin. This prowess is further accentuated by the model's parsimonious parameterization, boasting a mere 34M parameters, which not only contributes to its operational efficiency but also hints at untapped potential for further optimization. The discernible advantage of the DINO (OURS) model is underscored by its impressive balance between accuracy and efficiency, positioning it as a frontrunner in the domain of object detection. The illustrated AP curves in Fig. 7 serve as visual confirmation of these assertions, delineating the comparative performance of each algorithm and solidifying the superiority of the DINO (OURS) model in this critical task.

4.4 Ablation Study

In this comprehensive ablation study, we meticulously scrutinized the intricate interplay of various components within the DINO model to discern their individual contributions to overall performance. Initially, our investigations centered on the DINO model instantiated with the ResNet50 backbone, wherein the utilization of DC5 yielded a discernible average precision (AP) score of 58.1. Subsequent augmentation through the simultaneous incorporation of DC5, DCDN, and HKM techniques resulted in a marginal yet noteworthy refinement, elevating the AP metric to 58.3. This incremental enhancement underscores the subtle yet cumulative impact achieved through the amalgamation of these refined methodologies, further substantiating their efficacy in fortifying the model's precision. Consequently, we established the DINO-4scale configuration as the foundational baseline, leveraging the DC5 mode as a pivotal catalyst for subsequent analyses and optimizations.

The tabulated results from the ablation experiments, as delineated in Table 4, offer a comprehensive elucidation of the nuanced impacts of varying configurations on model performance. Notably, a pivotal shift occurs upon transitioning to the employment of the EfficientNetV2 backbone, manifesting in a discernible augmentation of the average precision (AP) metric from 59.5 to 60.2. This substantial improvement underscores the paramount importance of an adeptly engineered backbone network in bolstering detection efficacy. Particularly noteworthy is the confluence of EfficientNetV2 with the strategic integration of DC5, DCDN, and HKM methodologies, engendering not only a marked elevation in AP but also nuanced enhancements across distinct object size detection accuracies, as evidenced by improvements in AP50 and AP75. This holistic refinement underscores the symbiotic synergy achieved through the judicious fusion of advanced techniques, reaffirming their pivotal role in fostering balanced and comprehensive improvements in detection accuracy.

Table 4. Ablation results

Model	Backbone	DC5	DCDN	HKM	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params
DINO	Resnet50	✓			58.1	93.6	63.4	17.9	45.6	67.7	47M
	Resnet50	✓	✓	✓	58.3	93.3	64.5	17.6	43.2	67.3	47M
	Efficientnetv2	✓			59.5	**95.6**	62.8	18.9	**45.8**	68.9	**34M**
	Efficientnetv2	✓	✓	✓	**60.2**	94.5	**65.6**	**19.8**	45.2	**70.3**	34M

Moreover, a comparison of the number of parameters revealed that despite having fewer parameters in the Efficientnetv2 version (34M) compared to the Resnet50 version (47M), there was a significant performance improvement. This reflects the advantages of the Efficientnetv2 backbone in enhancing model efficiency and performance. Overall, this ablation experiment revealed the role of each technological component in enhancing the object detection performance of the DINO model, while also highlighting the importance of selecting a more efficient backbone architecture for optimizing model performance.

In our research, we implemented a methodology to introduce perturbations into bounding boxes, utilizing center shifting and box scaling techniques parameterized by the noise scale factors $\lambda 1$ and $\lambda 2$, respectively. This approach was meticulously designed to investigate the robustness of the bounding box predictions under varying degrees of noise. Specifically, our experimental design comprised two distinct series. In the first series, we kept the noise scale parameter $\lambda 2$ constant at a value of 2.0 while systematically adjusting $\lambda 1$ to evaluate its impact on the bounding box accuracy. Conversely, the second series of experiments involved maintaining $\lambda 1$ at a fixed level while modulating $\lambda 2$ to discern its influence on the bounding box performance. This structured exploration

of noise scales allowed us to comprehensively assess the sensitivity of our bounding box model to different types and magnitudes of noise, thereby enhancing our understanding of its resilience and generalizability in real-world applications.

A lower value of λ1 elicits only a marginal deviation in the centroid of the bounding box, thereby preserving the proximity of positional adjustments to their initial states. This refinement promotes the model's efficacy in precisely localizing objects but may not sufficiently fortify its resilience against spatial fluctuations. Our investigation underscores that the model achieves peak detection accuracy when λ1 is configured at 1.0 and λ2 at 2.0.

On the other hand, an increased λ1 value leads to more substantial center shifts, injecting greater positional diversity into the learning process. This diversity aids in enhancing the model's adaptability to changes in object location. Nonetheless, an excessive offset might cause the detected bounding box to deviate from the actual object, potentially compromising detection accuracy. The pertinent experimental results are presented in Table 5.

Table 5. Center shifting and box scaling

λ1	λ2	Epochs	AP	AP_S	AP_M	AP_L
0.25	2.0	50	58.2	18.5	45.1	67.7
0.5	2.0	50	59.7	19.1	45.7	68.8
1.0	2.0	50	**60.2**	**18.9**	**45.8**	**68.9**
2.0	2.0	50	58.4	17.7	44.6	68.7
1.0	1.0	50	58.5	17.6	45.2	67.4
1.0	2.0	50	**60.2**	**18.9**	**45.8**	**68.9**
1.0	3.0	50	59.6	19.2	45.6	68.9
1.0	4.0	50	58.1	19.1	45.0	68.7

For λ2, a lower value restricts the extent of box scaling, causing the model to primarily learn objects near their original scale. This condition favors the recognition of objects that are size-wise similar to those in the training dataset but might limit the model's flexibility to scale variations. Conversely, a higher λ2 value facilitates a wider range of box scaling, providing the model with extensive exposure to learning across varied object sizes. While this can improve the detection capabilities for objects of different sizes, an excessively large scaling range might lead to imprecise learning of object dimensions, consequently impacting the accuracy of detection.

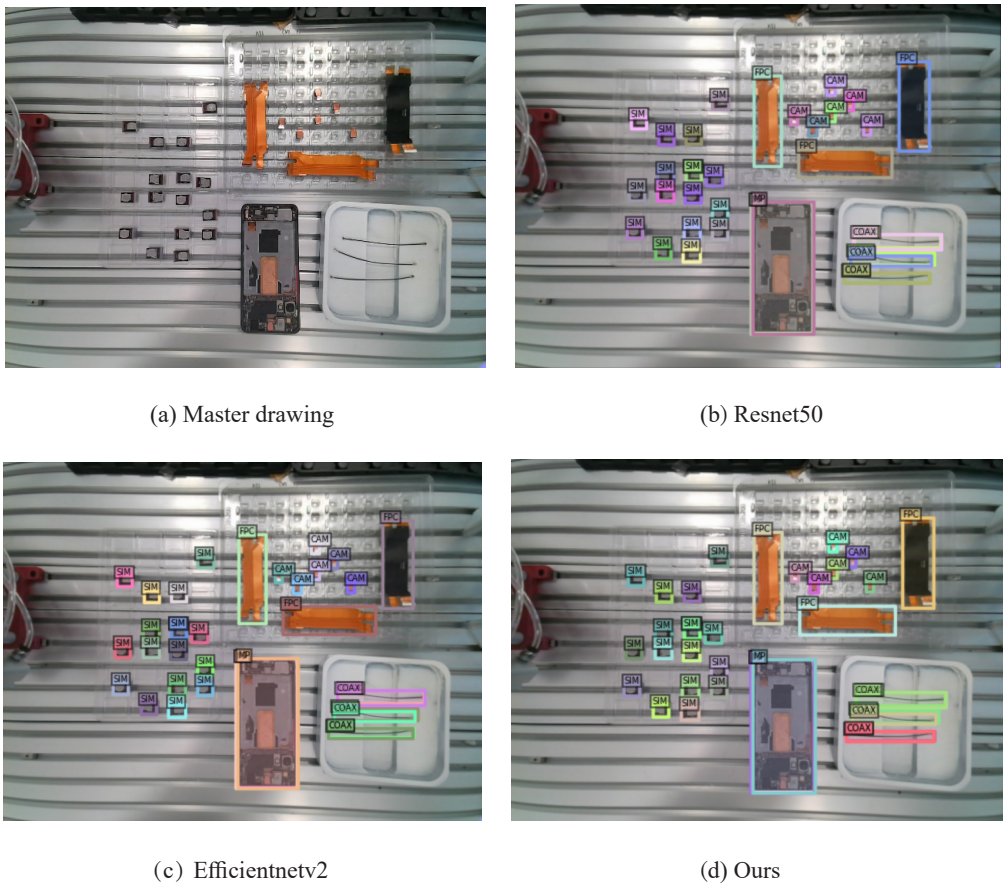
In the process of tuning the parameters λ1 and λ2, it is crucial to achieve a delicate equilibrium between the variability introduced by noise and the preservation of precise object features in the model. Striking the right level of noise is essential, as it can considerably enhance the model's ability to generalize and strengthen its robustness against diverse inputs. However, excessively high or low noise levels can detrimentally impact the model's effectiveness. Thus, the careful calibration of λ1 and λ2 is essential to optimize the performance of the object detection model, ensuring it delivers accurate and reliable results across varied scenarios. This requires a methodical approach to parameter tuning, guided by experimental data and informed by the specific characteristics of the objects being detected.

We continue to explore the impact of query denoising by altering the quantity of denoising queries, utilizing an optimized dynamic group. Performance significantly improves when the number of queries exceeds 50. However, after exceeding 100 denoising queries, further increasing the number of denoising queries only leads to a small additional performance improvement, and may even result in performance degradation. Ablation on number of denoising queries is shown in Table 6.

Table 6. Ablation on number of denoising queries

Denoising	50DCDN	50CDN	100CDN	40CDN	30CDN	20CDN	10CDN
AP	60.2	59.1	56.6	59.4	59.2	59.8	58.8

Additionally, the figures highlight the impact of each ablation component on key metrics such as accuracy, precision, and recall. Through visual analysis, insights into the algorithm's strengths and weaknesses under different configurations can be obtained. The effect diagram of 3C parts testing is shown in Fig. 8. This visual representation aids in making informed decisions regarding further optimizations and enhancements to the algorithm for better object detection performance.



(a) Master drawing (b) Resnet50

(c) Efficientnetv2 (d) Ours

Fig. 8. Detection results of four algorithms

5 Conclusion

This paper presents a target detection algorithm tailored for mobile assembly components within the 3C domain. Building upon DAB-DETR, we designate the queries in the decoder as dynamic anchor boxes and progressively refine them within the decoder layers. Following DN-DETR, it incorporates ground truth labels and noisy boxes in the decoder layers to stabilize bipartite matching during training. Building upon these foundations, a lightweight detection network architecture is proposed for mobile component assembly detection in the 3C scene, incorporating the Efficientnetv2 as the backbone model and utilizing the He Kaiming weight initialization method to extract robust feature maps, with training conducted using the efficient dynamic contrastive denoising method. Extensive results validate our algorithm's lightweight yet high-performance detection in the 3C scene.

6 Acknowledgement

This research was funded by National Natural Science Foundation of China (62276028), Major Research Program of National Natural Science Foundation of China (92267110), Beijing Natural Science Foundation Key Project (L233006) and Undergraduate Innovation Science and Technology Project of Universities in Beijing.

Reference

[1] L. Fu, Y. Zhang, Q. Huang, X. Chen, Research and application of machine vision in intelligent manufacturing, in: Proc. 2016 Chinese Control and Decision Conference (CCDC). IEEE, 2016.

[2] Z. Huang, Z. Yin, Y. Ma, C. Fan, A. Chai, Mobile phone component object detection algorithm based on improved SSD, Procedia Computer Science 183(2021) 107-114.

[3] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proc. of the IEEE conference on computer vision and pattern recognition, 2014.

[4] Z. Cai, N. Vasconcelos, Cascade R-CNN: Delving into high quality object detection, in: Proc. of the IEEE conference on computer vision and pattern recognition, 2018.

[5] J. Redmon, S. Divvala, R. Girshick, You only look once: Unified, real-time object detection, in: Proc. of the IEEE conference on computer vision and pattern recognition, 2016.

[6] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger in: Proc. of the IEEE conference on computer vision and pattern recognition, 2017.

[7] J. Redmon, A. Farhadi, Yolov3: An incremental improvement. <https://arxiv.abs/ 1804.02767>, 2018 (accessed 12.05.2020)

[8] T.Y. Lin, P. Goyal, R. Girshick, K. He, Focal loss for dense object detection, in: Proc. of the IEEE international conference on computer vision, 2017.

[9] M. Tan, R. Pang, Q.V. Le, Efficientdet: Scalable and efficient object detection, in: Proc. of the IEEE/CVF conference on computer vision and pattern recognition, 2020.

[10] N. Carion, F. Massa, G. Synnaeve, N. Usunier, End-to-end object detection with transformers, in: Proc. European conference on computer vision, 2020.

[11] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, J. Dai, Deformable detr: Deformable transformers for end-to-end object detection, in: Proc. of the International Conference on Learning Representations, 2022.

[12] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, Deformable convolutional networks, in: Proc. of the IEEE international conference on computer vision, 2017.

[13] Z. Sun, S Cao, Y. Yang, K.M. Kitani, Rethinking transformer-based set prediction for object detection, in: Proc of the IEEE/CVF international conference on computer vision, 2021.

[14] M. Tan, Q. Le, Efficientnetv2: Smaller models and faster training, in: Proc. International conference on machine learning. PMLR, 2021.

[15] D. Meng, X. Chen, Z. Fan, G. Zeng, Conditional detr for fast training convergence, in: Proc. of the IEEE/CVF international conference on computer vision, 2021.

[16] Y. Wang, X. Zhang, T. Yang, J. Sun, Anchor detr: Query design for transformer-based object detection, in: Proc. of the AAAI conference on artificial intelligence, 2022.

[17] X. Dai, Y. Chen, J. Yang, P. Zhang, L. Yuan, L. Zhang, Dynamic detr: End-to-end object detection with dynamic attention, in: Proc. of the IEEE/CVF international conference on computer vision, 2021.

[18] S. Liu, F. Li, H. Zhang, X. Yang, X. Qi, Dynamic anchor boxes are better queries for detr, in: Proc. of the International Conference on Learning Representations, 2022.

[19] F. Li, H. Zhang, S. Liu, J. Guo, L.M. Ni, L Zhang, Dn-detr: Accelerate detr training by introducing query denoising, in: Proc. of the IEEE/CVF conference on computer vision and pattern recognition, 2022.

[20] H. Zhang, F. Li, S Liu, L. Zhang, H. Su, Dino: Detr with improved denoising anchor boxes for end-to-end object detection, in: Proc. of the International Conference on Learning Representations, 2023.

[21] Z. Zong, G. Song, Y. Liu, Detrs with collaborative hybrid assignments training, in: Proc. of the IEEE/CVF international conference on computer vision, 2023.

[22] W. Lv, S. Xu, Y Zhao, G. Wang, J. Wei, C. Cui, Y. Du, Detrs beat yolos on real-time object detection, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, 2024.

[23] B. Koonce, B.E. Koonce, Convolutional neural networks with swift for tensorflow: Image recognition and dataset categorization, New York, NY, USA: Apress, 2021.

[24] D. Sinha, M. El-Sharkawy Thin mobilenet: An enhanced mobilenet architecture, in: Proc. 2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON), 2019.

[25] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proc. of the IEEE conference on computer vision and pattern recognition, 2018.

[26] P. Krähenbühl, C. Doersch, J. Donahue, T. Darrell, Data-dependent initializations of convolutional neural networks in: Proc. of the International Conference on Learning Representations, 2016.

[27] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proc. of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2010.

[28] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proc. of the IEEE international conference on computer vision, 2015.

[29] S. Gupta, M. Tan, Efficientnet-Edgetpu: Creating accelerator-optimized neural networks with automl. <https://ai. googleblog. com/2019/08/efficientnet-edgetpu-creating. html>, 2019 (accessed 10.11.2022).
[30] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proc. of the IEEE conference on computer vision and pattern recognition, 2018.
[31] J. Deng, W. Dong, R. Socher, L.J. Li, Imagenet: A large-scale hierarchical image database, in: Proc. 2009 IEEE conference on computer vision and pattern recognition, 2009.
[32] S. Elfwing, E. Uchibe, K. Doya, Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, Neural networks (107)(2018) 3-11.