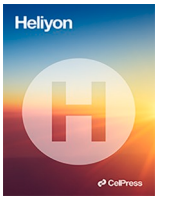
Heliyon 9 (2023) e19064

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon



# IYOLO-NL: An improved you only look once and none left object detector for real-time face mask detection

Yan Zhou [a,b,*]

[a] Ocean College, Zhejiang University, Zhoushan, 316021, China
[b] State Key Laboratory of Satellite Ocean Environment Dynamics, Second Institute of Oceanography, Ministry of Natural Resources, Hangzhou, 310012, China

ARTICLE INFO

*Keywords:*
YOLO
Label assignment
Anchor-free
Attention mechanism
Face mask detection

ABSTRACT

Object detection is a fundamental task in computer vision that aims to locate and classify objects in images or videos. The one-stage You Only Look Once (YOLO) models are popular approaches to object detection. Real-time monitoring of mask wearing is necessary, especially for preventing the spread of the COVID-19 virus. While YOLO detectors facing challenges include improving the robustness of object detectors against occlusion, scale variation, handling false detection and false negative, and maintaining the balance between higher precision detection and faster inference time. In this study, a novel object detection model called Improved You Only Look Once and None Left (IYOLO-NL) based on YOLOv5 was proposed for real-time mask wearing detection. To fulfill the requirement of real-time detection, the lightweight IYOLO-NL was developed by using novel CSPNet-Ghost and SSPP bottleneck architecture. To prevent any missed correct results, IYOLO-NL integrates the proposed PANet-SC with a multi-level prediction scheme. To achieve high precision and handle sample allocation properly, the proposed global dynamic-k label assignment strategy was utilized in an anchor-free manner. A large dataset of face masks (FMD) was created, consisting of 6130 images, for use in conducting experiments on IYOLO-NL and other models. The experiment results show that IYOLO-NL surpasses other state-of-the-art (SOTA) methods and achieves 98.8% accuracy while maintaining 130 FPS.

## 1. Introduction

Real-time and high-precision face mask detection is crucial for promoting epidemic prevention, as wearing masks is one of the most economical and effective methods for preventing COVID-19 infection. Object detection is a fundamental task in computer vision that aims to locate and classify objects in images or videos [1,2]. The recent progress in face mask detection research can be attributed to the development of deep learning techniques [3] and the availability of extensively annotated datasets [4]. Hand-crafted methods [5,6] and neural network methods [1,7] are two types of face mask detection methods based on the used features.

Hand-crafted methods rely on manually designed features [8], such as Haar-like [9], LBP [10], and HOG [11]. Dewantara et al. [12] used these features to train classifiers for detecting faces with different poses and occlusions, while He et al. [13] used skin color and eye features for mask wearing detection. However, hand-crafted methods have limited learning capacity and struggle to adapt to complex scenarios such as long distances and lighting changes.

* Ocean College, Zhejiang University, Zhoushan, 316021, China.
*E-mail address:* yanzhougemzhou@zju.edu.cn.

https://doi.org/10.1016/j.heliyon.2023.e19064
Received 6 June 2023; Received in revised form 9 August 2023; Accepted 9 August 2023
Available online 9 August 2023

Multi-stage methods involve at least two deep learning networks [14,15], generally including : human detection, face region detection, ROI extraction, feature vector extraction, normalization, and classification. However, the design of multi-stage methods is relatively complex, requires significant computational resources and expensive processing equipment.

Two-stage methods typically consist of face pre-detection and face category verification [16,17]. In the first stage, detectors provide feature descriptors for candidate faces. Face pre-detection is often achieved using multiple face detectors. The second stage consists of various classifiers aimed at determining mask-wearing status. Most two-stage methods combine a face detector with a classification model [18]. In many cases, the pre-detection model and classification model are trained separately [19], which can be time-consuming.

Single-stage methods have the largest share among neural network methods, including Faster *R*–CNN [2], SSD [20], YOLO series detectors [1,7,21,22], and other methods. Many studies have utilized You Only Look Once (YOLO) models for mask-wearing detection. Loey et al. [23] utilized the YOLOv2 detector with ResNet-50 for medical mask detection. A visualization system for YOLOv3 was implemented [24], which can achieve real-time inference on devices with low computational power and memory. Jiang et al. [25] improved YOLOv3 by using GIoU loss and focal loss to balance stability and robustness. Yu et al. [26] enhanced the YOLOv4 model by introducing a modified CSPDarkNet53 and applied it to the mask detection task. Yang et al. [27] used YOLOv5 for face mask detection, which outperformed other YOLO detectors while facing some technical difficulties and challenges.

While YOLOv5 is widely used in industry, it has limitations in detecting dense small faces with various types of masks and in handling facial occlusion issues in complex and volatile backgrounds [3,28]. Furthermore, it suffers high latency when used for real-time masked face inference.

The field of object detection research is constantly evolving, with researchers proposing novel architecture designs [3,29,30], loss functions [25], and data augmentation techniques [1,3,28]. Although various methods for reducing the size and computational cost of neural networks have emerged, including network pruning [31,32], low-bit quantization [33], and knowledge distillation [34,35], these methods are often limited by the constraints of the neural networks themselves. In contrast, building deep neural networks with fewer parameters and more efficient computations shows greater potential than relying on complex parameter adjustment [3].

The mentioned studies and frameworks suffer from the same drawbacks, achieving satisfactory results for large-simple objects, but not robust enough for scene-specific, small, and occluded objects. This creates challenges in detecting masked faces due to low resolution and limited appearance information. To address the problems of conventional detectors in face mask detection, this paper proposes an IYOLO-NL detection model.

The main contributions of the paper are summarized as follows.

- This paper introduces an anchor-free approach to the YOLO model and constructs several lightweight real-time network structures. The computationally and time-intensive anchor-based manner is replaced by a more efficient approach where anchor points are rapidly selected and directly regressed to target objects. Additionally, novel light CSPNet-Ghost bottleneck and SSPP bottleneck are utilized in IYOLO-NL to extract features more effectively and reduce inference lag.
- To address the problem of insufficient feature information for multi-scaled and scene-varied objects, we propose a self-attention PANet-SC neck. In combination with a multi-level prediction scheme, the developed neck continuously fuses feature maps, narrows the semantic gap, and strengthens multi-scale features to enhance prediction ability.
- To address the problem of assigning ambiguous samples and improve the classification and localization accuracy of occluded objects, a novel global dynamic-k label assignment strategy has been proposed. The strategy is launched through decoupled heads to avoid complex anchor box operations, and displacement problems.
- In this paper, we construct a large face mask dataset and evaluate the performance of the proposed IYOLO-NL through several experiments, including a comparative analysis with related models.

The paper is structured as follows: Section 2 provides a detailed review of the related work. Section 3 presents a detailed description of the proposed IYOLO-NL model, including its anchor-free approach, inference acceleration, and label assignment method. In Section 4, a series of experiments are conducted to compare and validate the performance of the proposed IYOLO-NL model with SOTA algorithms. Finally, Section 5 summarizes the content of the paper.

## 2. Related work

This section outlines the latest enhancements to the backbone, neck, and head components of the current YOLO series models. It then provides a detailed summary of the YOLOv5 model's workflow and limitations.

### *2.1. Anchor-free detectors*

Anchor-free detectors [3,28,29,36] have experienced significant development in the past two years, and several studies have demonstrated that detection models based on anchor-free strategies achieve comparable performance to anchor-based methods [28, 37]. Anchor-based detectors utilize anchor boxes of varying shapes and sizes as training samples and determine their labels based on the IoU between the anchor boxes and ground-truth bounding boxes. In contrast, anchor-free methods regress directly to the ground-truth bounding boxes, utilizing anchor points in the input image as training samples.

Moreover, the anchor-free mechanism significantly reduces the number of heuristic hyperparameters that must be tuned, avoiding the need for complex training techniques such as clustering and grid sensitivity [28,38]. This approach delivers superior performance

while reducing computational complexity.

### 2.2. YOLO CSPNet backbone design

Cross Stage Partial Networks (CSPNet) is an efficient CNN architecture [30] that is used as a backbone in object detection models. It works by dividing the feature maps from a previous layer into two parts. One part goes directly to the next stage, and the other part passes through convolutional layers before being merged back. This design reduces computation and memory costs while maintaining or even enhancing the representational capacity of the network.

YOLOv4 [1,39] and YOLOv5 utilize CSPDarknet53 as the backbone network, while YOLOX [28] employs CSPNet and other advanced techniques to balance detection speed and accuracy.

### 2.3. YOLO neck design

The Feature Pyramid Network (FPN) is a top-down architecture [40] that leverages previously learned features from a backbone network and integrates them with new features using lateral connections. This approach facilitates the detection of objects at varying scales and resolutions, which is especially advantageous for identifying small objects. YOLOv5 utilizes FPN to extract features from various levels of the backbone network and create multi-scale feature maps that enhance object detection. However, aligning anchor boxes with objects of different sizes on FPN feature maps presents significant challenges. The anchor box size is dataset-specific and varies as the data changes. The predicted box size on each feature map should depend on the feature map's receptive field [37], i.e., the network structure itself.

The Progressive Attention Network (PANet) [41] utilizes a bottom-up architecture that incorporates features from different levels of the backbone network by means of a lateral aggregation module (LAM) for each level. This design captures both fine-grained and coarse-grained features [3,39], enabling the network to detect both small and large objects.

### 2.4. Label assignment strategy

The FCOS network [37] considers any point inside the ground-truth bounding box as a positive sample, but it fails to handle ambiguous samples properly. Only assigning ambiguous sample points to the smallest ground-truth bounding box leads to poor detection performance for overlapping objects [28,42,43].

From ATSS [44] and PAA [45] to Auto Assign [46], researchers have made efforts to enhance the flexibility of label assignment. However, these methods only explore the optimal allocation strategy for individual objects and do not consider contextual information from a global perspective. Additionally, a significant number of ambiguous samples are discarded without being fully utilized.

Therefore, a better allocation strategy should aim for global optimality. YOLOX [28] and OTA [43] view label assignment as an optimal transport problem and globally optimize the label assignment for object samples in the image. However, their simple approach of treating specific points as positive samples lead to an excessive number of low-confidence samples [42]. Consequently, the computational efficiency is not significantly different from anchor-based mechanisms, leading to slow real-time inference.

### 2.5. YOLOv5 model detection workflow

YOLOv5 is a one-stage detection model, like previous YOLO detectors [1,7,39], that can be divided into three parts: backbone, FPN neck, and YOLO head. The workflow of the YOLOv5 is summarized as follows: feature extraction, feature enhancement, and predicting the object situation.

To be specific, YOLOv5's backbone is CSPDarknet, which extracts three effective feature layers as the following steps.

- The first layer of backbone is Focus. The Focus network captures values at every other pixel in an image, then the resulting four independent feature layers are stacked to expand the channel dimension. The concatenated feature layers consist of 12 channels, as opposed to the original three channels.
- The following CSPDarknet consists of CSPLayer and SPP bottleneck. The CSPLayer comprises a backbone of $1 \times 1$ and $3 \times 3$ convolutions, along with a residual edge connection. Residual blocks in the network leverage skip connections to avoid gradient vanishing in deep neural networks. YOLOv5 uses SiLU as its activation function, which is an improved version of the Sigmoid and ReLU functions. After each CSPLayer, a BN layer and a SiLU activation function are stacked.
- The Spatial Pyramid Pooling (SPP) bottleneck structure [47] is then used for feature extraction via maximum pooling with different kernel sizes. In YOLOv4, SPP was used in FPN, while in YOLOv5, the SPP module is integrated into the backbone feature extraction network. The backbone ultimately extracts three effective feature layers, with shapes of (80, 80, 256), (40, 40, 512), and (20, 20, 1024), respectively.
- Then, the FPN network fuses features from different layers to enhance feature extraction. The neck output comprises three enhanced features: (20, 20, 1024), (40, 40, 512), and (80, 80, 256).
- The feature layers consist of sets of feature points. The coupled head in YOLOv5 is responsible for detecting whether each feature point corresponds to an object. Classification and regression are performed through a single $1 \times 1$ convolution, much like in previous YOLO versions.

However, YOLOv5 exhibits a significant misalignment problem [3,28]. While its approach mitigates gradient disappearance in deep neural networks and is widely used in industry, its detection accuracy is low during inference and difficult to improve. Based on the above studies, this paper summarizes the possible reasons for the low detection accuracy of YOLOv5.

- The performance improvement of YOLOv5 mainly results from optimization and fine-tuning, primarily relying on anchor-based mechanisms and manually matched rules. It is relatively independent of the development in the object detection research field. Anchor-based mechanisms in YOLOv5 generally require the specification of the origin coordinates and box size, necessitating anchor-based translation during prediction. The tuning process in YOLOv5 involves optimizing the anchor box, which is challenging due to different datasets requiring distinct hyperparameter adjustments.
- Training and inference times for YOLOv5 are lengthy, rendering real-time object detection on complex datasets unfeasible. Additionally, it exhibits suboptimal utilization of GPU computational resources, with high computational and time complexity.
- YOLOv5 fails to handle blurry and multi-scale objects in sufficient detail. Its manual rule-setting strategy neglects precise matching for small objects, making it ineffective in diverse scenarios. The excessive use of normalization leads to indistinct learned features, making it challenging to classify targets of varying scales.
- The coupled head in YOLOv5 performs classification and regression tasks simultaneously. However, during inference stage, misalignment between the regression and classification branches leads to low detection accuracy.

## 3. Methodology

To achieve fast and lightweight computation, the YOLO object detector was transformed into anchor-free manner. The backbone was reconstructed by deploying the proposed CSPNet-Ghost bottleneck, Serial Spatial Pyramid Pooling network. To enhance feature extraction and achieve high-precision detection for multi-scaled objects, the self-attention PANet-SC was developed with multi-level prediction scheme in the neck section. To handle ambiguous and complicated sample points and promote IYOLO-NL to state-of-the-art level, the global dynamic-k label assignment strategy is proposed. The decoupled head is utilized with optimized cost loss functions. The structure of IYOLO-NL model is illustrated in Fig. 1.

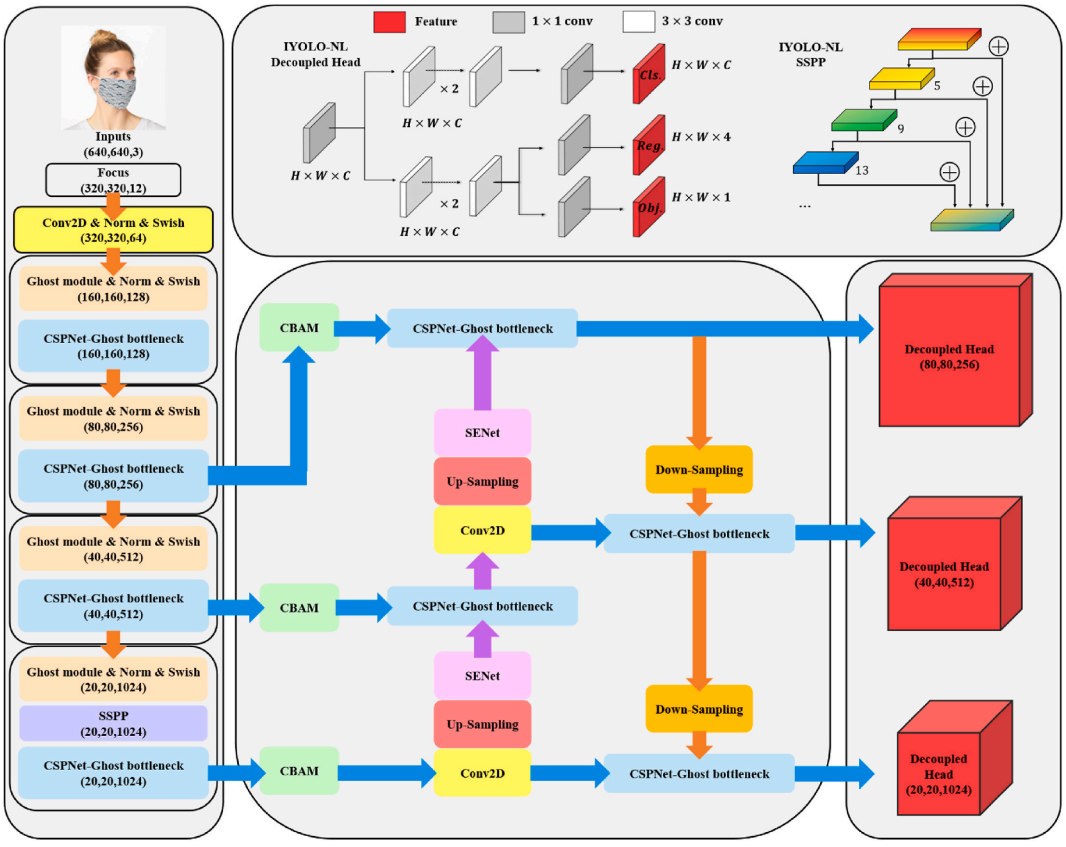



**Fig. 1.** IYOLO-NL model structure. The IYOLO-NL model comprises a newly improved backbone, neck, and head.

### 3.1. Rapid lightweighting backbone of IYOLO-NL

This subsection presents an extensive description to anchor-free methods, along with the novel CSPNet-Ghost bottleneck and SSPP structure, which are proposed to optimize GPU computing resources and accelerate YOLO inference speed.

#### 3.1.1. Anchor-free IYOLO-NL

The YOLO object detection process was redefined in an anchor-free manner using the per-pixel prediction method [28,37,38]. We denote the feature map in the i-th layer of the backbone network as $F_i \in \mathbb{R}^{H\times W\times C}$. The ground-truth bounding boxes of the input image are defined as $\{B_i\}$ Eq. (1),

$$B_i = \left(x_0^{(i)}, y_0^{(i)}, x_1^{(i)}, y_1^{(i)}, c^{(i)}\right) \in \mathbb{R}^4 \times \{1, 2, \ldots, C\} \tag{1}$$

where, $(x_0^{(i)}, y_0^{(i)})$ and $(x_1^{(i)}, y_1^{(i)})$ represent the coordinates of the upper-left and lower-right corners of the i-th bounding box. $c^{(i)}$ denotes the class attribute of the object in the bounding box, and C is the number of classes.

The coordinate $\left(\left\lfloor\frac{s}{2}\right\rfloor + xs, \left\lfloor\frac{s}{2}\right\rfloor + ys\right)$ can be used to map anchor point $(x, y)$ from feature map $F_i$ back to its corresponding position in the original image. Here, s is the total stride from the original image to the current feature map. The central area of the ground-truth bounding box with center coordinate $(c_x, c_y)$ is defined as $(c_x - rs, c_y - rs, c_x + rs, c_y + rs)$, where r is a hyperparameter set to 1.5 for the COCO dataset [3,22].

Anchor points within the central area were considered positive samples with label category $c^*$ equal to that of the ground-truth bounding box they correspond to, while the rest ones were treated as negative samples with $c^* = 0$, indicating the background samples. As depicted in Fig. 2, each anchor point corresponds to a four-dimensional real vector $v^* = (l^*, t^*, r^*, b^*)$ representing its regression parameter [37] Eqs. (2) and (3).

$$l^* = \frac{x - x_0^{(i)}}{s}, \quad t^* = \frac{y - y_0^{(i)}}{s} \tag{2}$$

$$r^* = \frac{x_1^{(i)} - x}{s}, \quad b^* = \frac{y_1^{(i)} - y}{s} \tag{3}$$

where $x_0^{(i)}$, $y_0^{(i)}$, $x_1^{(i)}$, $y_1^{(i)}$ denote the left, top, right and bottom locations of ground-truth bounding box, respectively.

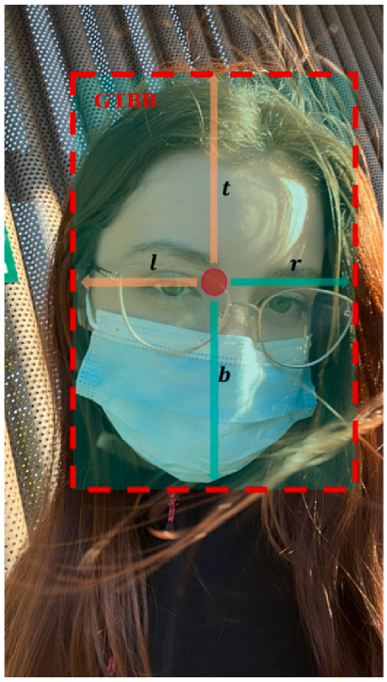



**Fig. 2.** Schematic diagram of anchor-free regression pattern. The red dashed box represents the ground-truth bounding box (GTBB), the orange solid arrow represents the left and top distances from the anchor point to the bounding box, while the blue solid arrow represents the right and bottom distances.

### 3.1.2. CSPNet-Ghost bottleneck design

To achieve satisfactory accuracy, YOLOv5 demands significant floating-point computing and storage resources for traditional convolutional neural network (CNN) operations (Fig. 3 (a)). Specifically, generating a feature map from a 640 × 640 image requires 4.75 billion floating-point operations for YOLOv5. Although the feature maps connected by red dotted lines in Fig. 3 (c) have high similarity, the corresponding redundant feature vectors reveal essential semantic information hidden in the image, which is essential for high-precision object detection.

To reduce the computational cost of generating similar feature maps (i.e., ghost feature maps), the backbone was optimized by utilizing the proposed CSPNet-Ghost bottleneck network. The CSPNet-Ghost bottleneck comprises two ghost modules. The ghost module [48] employs smaller convolution kernels (e.g., 3 × 3) to generate feature maps for its operations (Fig. 3 (b)). Specifically, a set of m feature maps $Y \in \mathbb{R}^{h\times w\times m}$ is obtained from the original image using basic convolution operations Eq. (4):

$$Y = X * f \tag{4}$$

$f \in \mathbb{R}^{c\times k\times k\times m}$ is the convolution kernel, $m \leq n$. The output feature map's spatial dimensions (height and width) are made consistent with those generated by convolutional operations by keeping the parameters identical. Then, a series of linear operations is applied to Y to generate ghost feature maps, resulting in the final n feature maps Eq. (5).

$$y_{ij} = \Phi_{i,j}(y_i)\ \forall i = 1,\ldots,m\ \ \forall j = 1,\ldots,r \tag{5}$$

here y is the i-th feature map in Y, has s ghost feature maps $\{y_{ij}\}_{j=1}^{r}$. The linear operation $\Phi_{i,j}$ maps y to ghost feature map $y_{ij}$. $\Phi_{i,r}$ is the identity mapping with respect to Y. The set of ghost feature maps generated by the ghost module is denoted as $[y_{11}, y_{12}, \ldots, y_{mr}]$.

To expedite the training and inference speed of YOLO backbone, the CSPNet-Ghost bottleneck was developed. Ghost and SENet attention modules [49], illustrated in Fig. 3 (d), were utilized in backbone part for efficient feature extraction. Consistent feature map dimensions between input and output were maintained by using depthwise separable and 1 × 1 convolution in CSPNet-Ghost bottleneck's residual edge.

### 3.1.3. Serial Spatial Pyramid Pooling network

Training deeper neural networks is an arduous task. To tackle this challenge, a novel Serial Spatial Pyramid Pooling network (SSPP) was proposed, drawing inspiration from residual learning [50]. As shown in Fig. 4 (a) and (b), max pooling operations of different sizes are concatenated vertically. The residual edge transfers the feature maps from the upper CSPNet-Ghost bottleneck outputs without modification. Finally, all the feature maps are aggregated together.

## 3.2. Multi-level self-attention neck of IYOLO-NL

This subsection presents a systematic introduction to the attention mechanism, the proposed PANet-SC neck structure, and the multi-level prediction scheme.

### 3.2.1. Attention mechanisms

Attention mechanisms play a critical role in modern object detection models [39,49,51,52]. These mechanisms enable the model to focus on specific parts of an input image, which is useful for detecting small or partially occluded objects. The core of the attention mechanism is to allow convolutional neural networks to adaptively attend to significant objects.

Various forms of attention mechanisms are used in object detection. As illustrated in Fig. 5 (a), Squeeze-and-Excitation Networks (SENet) [49] enhance important features and improve accuracy by utilizing the correlation between channel features and won the 2017 ILSVR competition. The Convolutional Block Attention Module (CBAM) [51] integrates both channel and spatial attention mechanisms to promote the performance of convolutional neural networks (Fig. 5 (b)). Channel attention focuses on "what" objects are

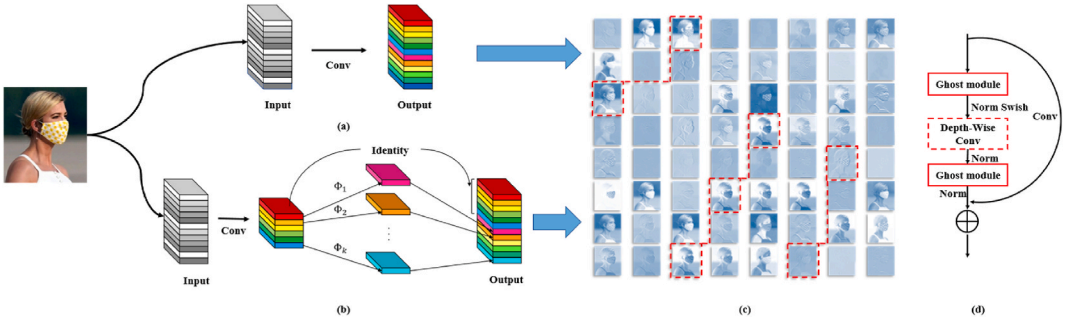



**Fig. 3.** Efficient feature map processing through CSPNet-Ghost bottleneck. (a) Traditional convolution operation. (b) Ghost module operation. With upper and lower parts generated by convolution and linear operation, respectively. (c) Feature maps from traditional convolution or ghost module. (d) CSPNet-Ghost bottleneck.

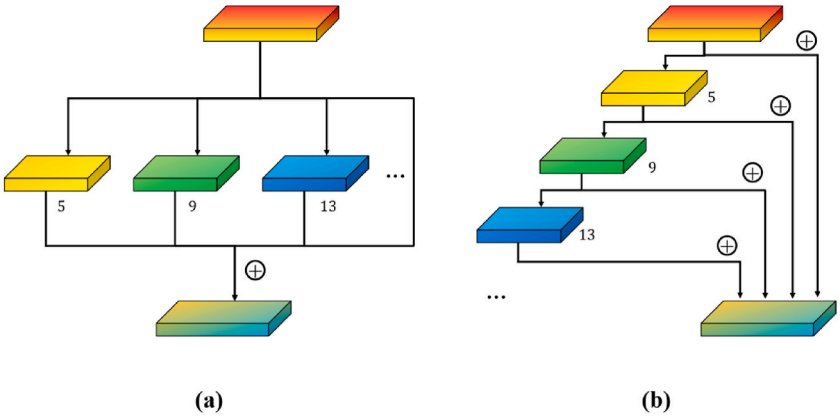



**Fig. 4.** The traditional and novel Spatial Pyramid Pooling network. (a) SSP. (b) SSPP.

of observational significance, while spatial attention focuses on "where" meaningful objects are located.

Overall, attention mechanisms have been demonstrated to significantly enhance the performance of object detection models, particularly in challenging scenarios where objects are small, occluded, or have complex backgrounds.

#### 3.2.2. Multi-level prediction scheme

Fig. 6 (a) and (b) illustrates our proposed PANet-SENet-CBAM (PANet-SC) network architecture. To be specific, $B_1 \sim B_3$ represent feature maps in the backbone, $P_1 \sim P_3$ represent unsampled feature maps in the neck, and $H_1 \sim H_3$ represent concatenated feature maps after downsampling [40,41]. The green dash curved arrow shows the ability of the SENet attention network to capture key points across channels, while the blue dash curved arrow represents the ability of the CBAM attention network to capture key points across channels and spaces.

In IYOLO-NL, the anchor-free approach eliminates the need for manual design of anchor box. Our model also employs a multi-level prediction scheme [37] across different feature map levels to enable object detection at varying scales. The specific method is as follows.

- Calculate the regression target parameters $l^*$, $t^*$, $r^*$, and $b^*$ for each pixel in the feature map.
- If the anchor points of a certain $F_i$ meet the condition Eq (6):

$$\begin{cases} \max(l^*, t^*, r^*, b^*) \leq m_{i-1} \\ \max(l^*, t^*, r^*, b^*) \geq m_i \end{cases} \tag{6}$$

then set the pixel as a negative sample. Here $m_i$ is the maximum distance that needs to be regressed for feature map i. In the materials and experiments section (section 4) of this paper, the maximum regression distances $m_1$, $m_2$, and $m_3$ for the fused feature maps $H_1 \sim H_3$ are set to 16, 32, and 64, respectively.

### 3.3. Global dynamic-k label assignment strategy

In this subsection, we proposed global dynamic-k label assignment strategy, through Big Sieve and Small Sieve algorithm to allocate anchor points in an effective manner. Fig. 7 illustrates the schematic diagram of global dynamic-k label assignment strategy.

#### 3.3.1. Initial assignment - Big Sieve

The Big Sieve algorithm is proposed to find all candidate anchor points in a specified area and initially divide positive samples. The algorithm calculates relative distance to determine the mask attribute in the specified area.

As depicted in Fig. 8 (a) and (b), two mask sets $\mathscr{B}^*$ and $S^*$ are derived, with $\mathscr{B}^*$ denoting the mask set for the anchor points in the ground-truth bounding box and $S^*$ signifying the mask of the anchor points in the subdomain $\pi^*$. The intersection and union operations are executed on these sets, yielding Eq. (7):

$$\begin{cases} \text{Full} = \mathscr{B}^* \cup S^* \\ \text{Inner} = \mathscr{B}^* \cap S^* \end{cases} \tag{7}$$

here, Full mask set indicates that anchor points either within the ground-truth bounding box or inside the central subdomain $\pi^*$ range. In the Inner mask set, a True flag implies that the corresponding anchor point lies inside the center of the ground-truth bounding box. True anchor points are potential candidate positive samples, while anchor points with False masks are excluded and assigned a large cost weight [3,28], typically 100,000.

In the initial assignment stage, adjacent anchor points with higher similarity may share the same affiliation. To filter candidate

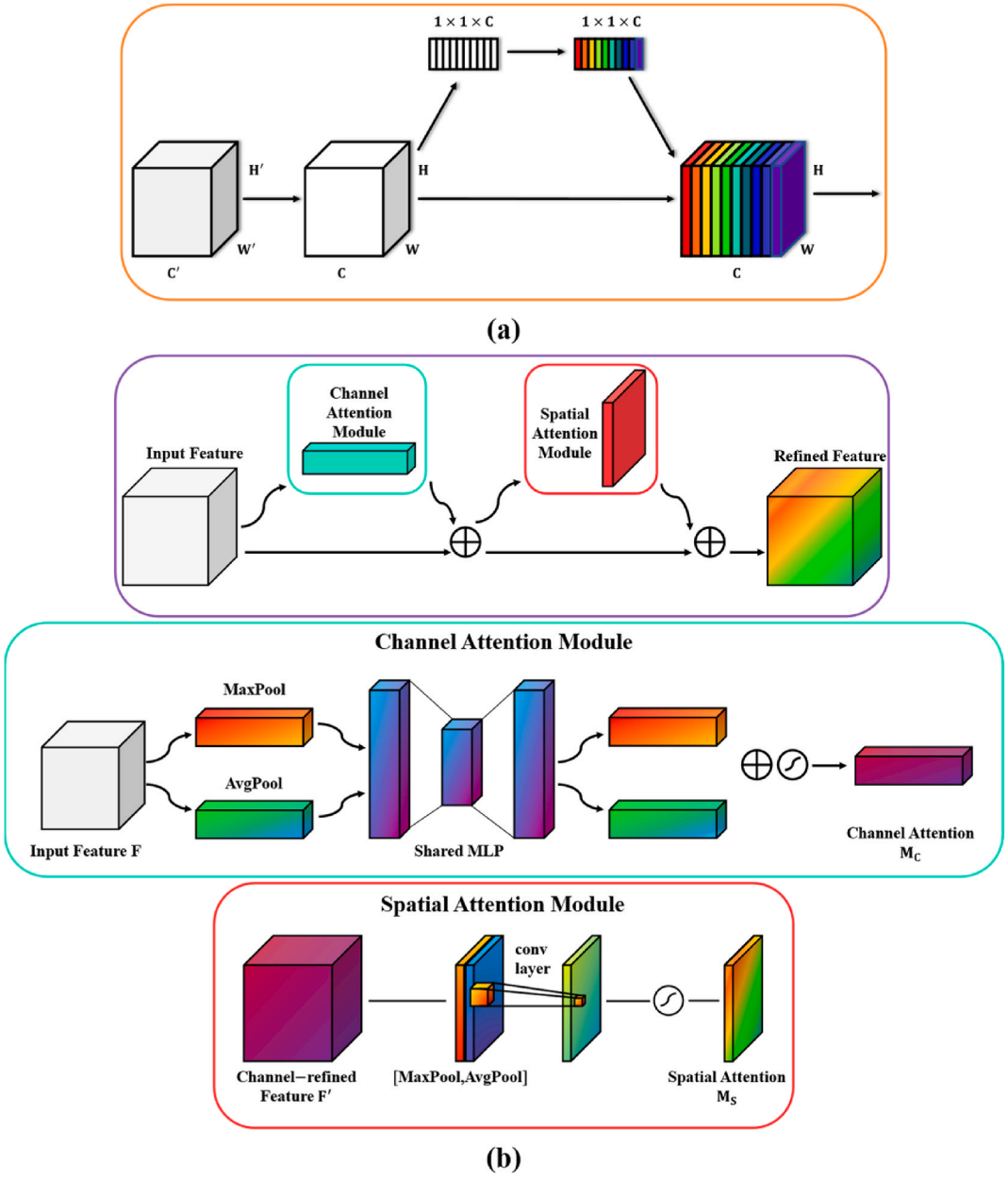



**Fig. 5.** Schematic diagrams of attention mechanisms. (a) SENet. (b) CBAM.

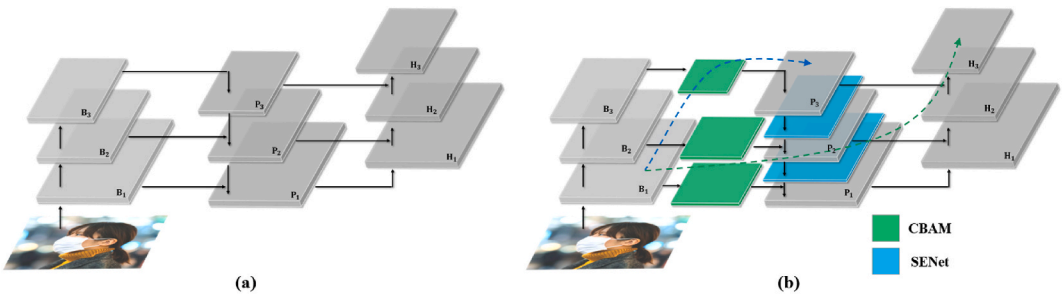



**Fig. 6.** The traditional PANet and novel PANet-SC network structure. (a) The PANet network. (b) The proposed PANet-SC network.

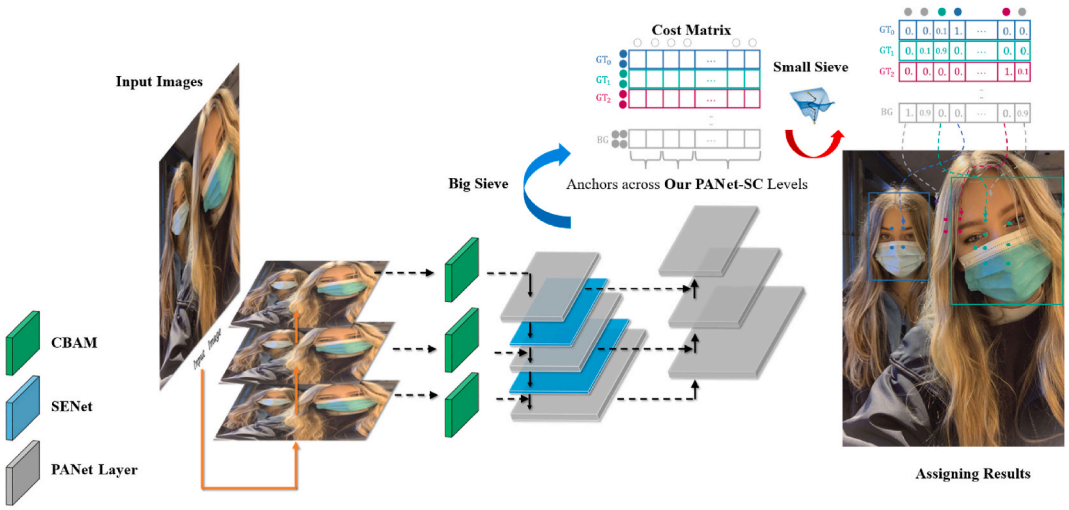



**Fig. 7.** The proposed global dynamic-k label assignment strategy.

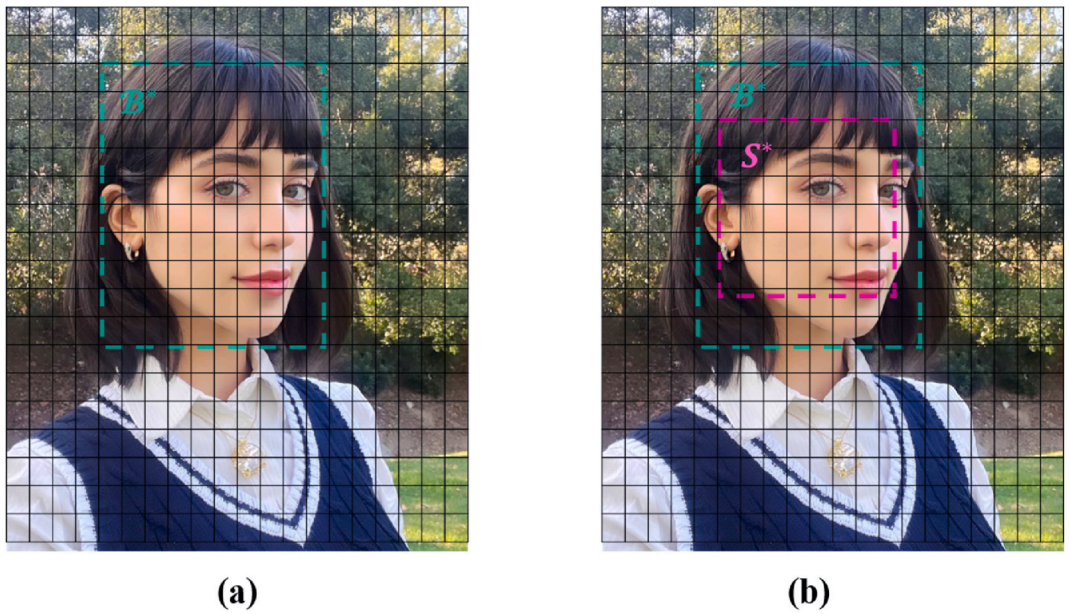



**Fig. 8.** Selection of area using the Big Sieve algorithm. The light green dotted box in (a) is the ground-truth bounding box, and the pink dotted box in (b) is the central region $\pi^*$ (subdomain) generated by the midpoint of the ground-truth bounding box.

samples during the initial screening for each object, the Big Sieve algorithm starts from the midpoint of the ground-truth bounding box and selects points in $\pi^*$ every other point (stride equals to 1) until reaching the boundary of Inner set. This significantly reduces the number of redundant candidate points compared to YOLOX [28] and FCOS [37], thereby decreasing the likelihood of low-confidence samples.

### 3.3.2. *Fine-grained assignment - Small Sieve*

To achieve fine-grained assignment, the cost matrix is computed between ground-truth bounding boxes and candidate prediction boxes generated by candidate anchor points. The GIoU function is used as the bounding box loss Eq. (8):

$$L_{reg} = -\log\left(\mathrm{GIoU}\left(B_{gtbb}, B_{pred}\right)\right) \tag{8}$$

where $B_{gtbb}$ and $B_{pred}$ represent the ground-truth bounding box and candidate prediction boxes, respectively. The bounding box loss function quantifies the dissimilarity between $B_{gtbb}$ and $B_{pred}$.

The class branch loss is calculated using a binary cross-entropy function that incorporates the Obj prediction branch Eq. (9):

$$L_{cls} = -\sum_{i=1}^{n}(t_i \log(p_i) + (1 - t_i)\log(1 - p_i)) \tag{9}$$

where $p = [p_0,\ldots,p_{c-1}]$, $p_c$ represents the probability that the sample is predicted as the c-th class. In Obj prediction branch, t represents the label of the sample Eq. (10),

$$t = \begin{cases} 1 & \text{Positive Sample} \\ 0 & \text{Negative Sample} \end{cases} \tag{10}$$

Therefore, the cost matrix can be obtained as follows [3,28,36] Eq. (11):

$$c_{ij} = L_{ij}^{cls} + \lambda L_{ij}^{reg} \tag{11}$$

where $\lambda$ is generally set to 3, $c_{ij}$ represents the cost matrix of all candidate anchor points. As the value of $c_{ij}$ decreases, the matching degree between the sample point and a certain ground-truth bounding box increase.

For each ground-truth bounding box, the proposed Small Sieve algorithm is utilized to dynamically select k precise positive sample points from the preliminary candidate anchor points identified by the cost matrix. The algorithm pseudocode is provided in Appendix.

In the fine-grained assignment stage, the Small Sieve algorithm guarantees that each ground-truth bounding box has k anchor points. Additionally, each ambiguous sample point is associated with a unique globally optimal ground-truth bounding box.

The schematic diagram of the Big Sieve and Small Sieve algorithm is demonstrated in Fig. 9(a–h). Following the Big Sieve and Small Sieve assignments, at least one prediction box is generated for each object, with each prediction box corresponding to a unique ground-truth bounding box.

### 3.3.3. *Decoupled head of IYOLO-NL*

The YOLOv5 coupled head, which uses a $1 \times 1$ convolution operation for classification and regression, suffers from misalignment issues [3,28,37]. To tackle this challenge, we implemented the decoupled head from YOLOX to separately handle localization and regression [28]. By avoiding misalignment, the decoupled head significantly improves the overall detection performance.

Using the anchor-free method and the proposed global dynamic-k label assignment strategy, k anchor points are assigned to each ground-truth bounding box. However, during the final inference testing, only the most accurate predicted bounding box needs to be

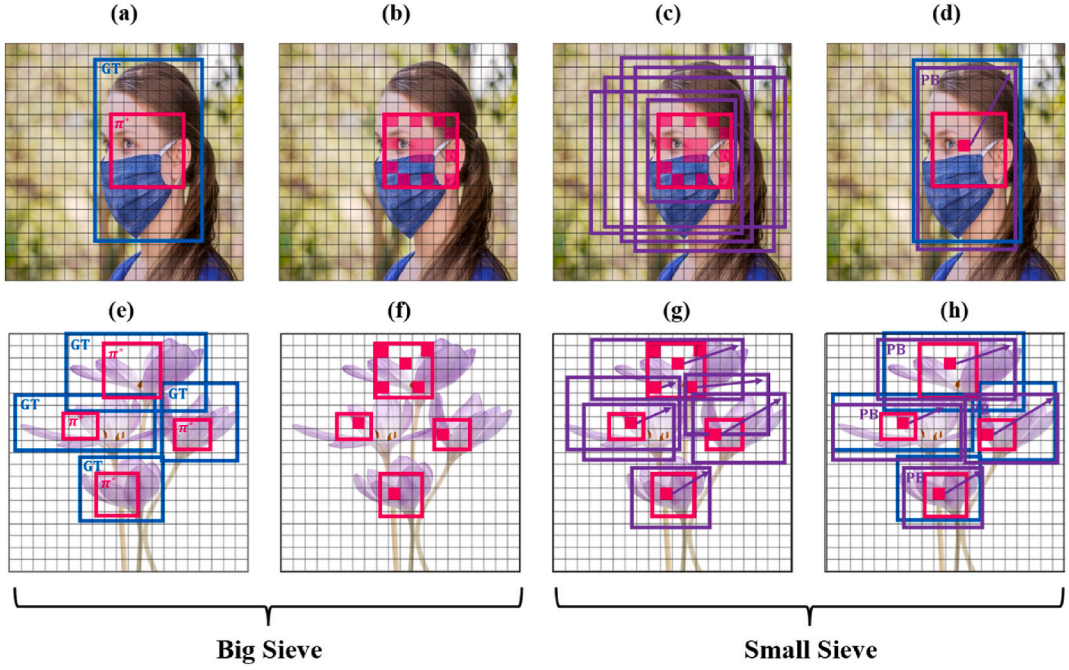



**Fig. 9.** The overall process of the global dynamic-k label assignment strategy. (a), (b), (c), (d) represent the single object condition. (e), (f), (g), (h) represent the dense and overlaying objects condition. The blue bounding boxes in (a), (e) represent the ground-truth bounding boxes, the pink bounding boxes, and blocks in (b), (f) represent the regions and candidate sample points obtained after the initial screening by Big Sieve. The purple bounding boxes in (c), (g) represent the prediction boxes generated by the candidate sample points. The (d), (g) represent that each object has k candidate points (take 1 as an example), and the corresponding candidate prediction box matches with the related object's ground-truth bounding box. The purple arrows in (d), (g), (h) represent the optimal cost matrix vector for each candidate sample points. For the sake of clarity, the purple arrows are omitted in (c), and let objects' k equals to 1 in (d), (h).

matched with each object in the image or video. Thus, the number of anchors associated with each ground-truth bounding box is reduced from k to 1. To achieve this, the IYOLO-NL algorithm incorporates the Non-Maximum Suppression (NMS) algorithm, which eliminates redundant predicted bounding boxes and retains the most accurate ones.

## 4. Materials and experiments

In this section, we constructed a face mask dataset from scratch. A series of ablation and comparative experiments were then conducted to validate the improved performance of the proposed IYOLO-NL over the baseline model. In addition, the real-time detection performance of the IYOLO-NL algorithm was evaluated by deploying multiple YOLO models and other state-of-the-art (SOTA) detectors.

### 4.1. The face mask dataset

To verify the detection capability of IYOLO-NL on complex scenes and overlapping objects. We created a comprehensive face mask dataset (FMD) comprising of 6130 images, as detailed in Table 1. The dataset includes 29,569 manually annotated ground-truth bounding boxes for 6130 pictures, covering 12 distinct scenarios. As demonstrated in Fig. 10(a–d), the horizontal-vertical flipping and hue-saturation-exposure adjustment techniques were applied to augment the dataset, resulting in 18,390 images. The dataset was divided into three parts, 70% for training, 20% for validation while the remainder was used for testing. The valid datasets were divided into 10 parts, with each part containing 368 images except for the last two parts, which consisted of 367 images each. For the test datasets, images with dense and tiny-sized faces were classified as small objects (714 images), images with one or two faces were classified as large objects (494 images), and the remaining images were classified as medium objects (631 images). All images were labeled as three categories.

- With mask
- Without mask
- Mask worn incorrectly

### 4.2. Experimental settings

To improve the model's detection capability for multiscale objects, mosaicking and mix-up [53] augmentations were employed during the training phase. Table 2 presents the experimental environment configuration for all models in this paper, and Table 3 illustrates that all models were trained for 100 epochs until maximum accuracy was achieved. The first 5 epochs were dedicated to the pretraining stage, and formal training began at the 6th epoch. Mosaic augmentation was turned off in the last 15 epochs.

### 4.3. Evaluation indicators

The common evaluation metrics are: Accuracy, Precision ($P_c$), Recall ($R_c$), Average Precision (AP), and mean Average Precision (mAP), which are defined as follows Eqs. (12), (13), (14), (15) and (16):

$$\text{Accuracy} = \frac{TP_c + TN_c}{TP_c + TN_c + FP_c + FN_c} \tag{12}$$

$$P_c = \frac{TP_c}{TP_c + FP_c} \tag{13}$$

**Table 1**
The face mask dataset. Providing the number of images (Number of Pics) for each background class, along with the corresponding number of manually annotated ground-truth bounding boxes (Number of GTBBs).

| S/N | Background Environment | Number of Pics | Number of GTBBs |
|---|---|---|---|
| **1** | Car, Train, Airplane | 660 | 4682 |
| **2** | Road, Avenue, Booth | 800 | 6565 |
| **3** | Grasses and Flowers | 600 | 2113 |
| **4** | TV interview, Tik Tok | 720 | 1819 |
| **5** | Dog, Cat, Horse | 300 | 330 |
| **6** | Rainy, Cloudy, Snow Day | 450 | 585 |
| **7** | Dim light | 400 | 649 |
| **8** | Family portrait, Many people | 600 | 5195 |
| **9** | Selfie | 200 | 312 |
| **10** | Object or Human occlusion | 500 | 522 |
| **11** | Concert | 550 | 4526 |
| **12** | Canteen, Hospital | 350 | 2271 |
| **Total** | | **6130** | **29,569** |

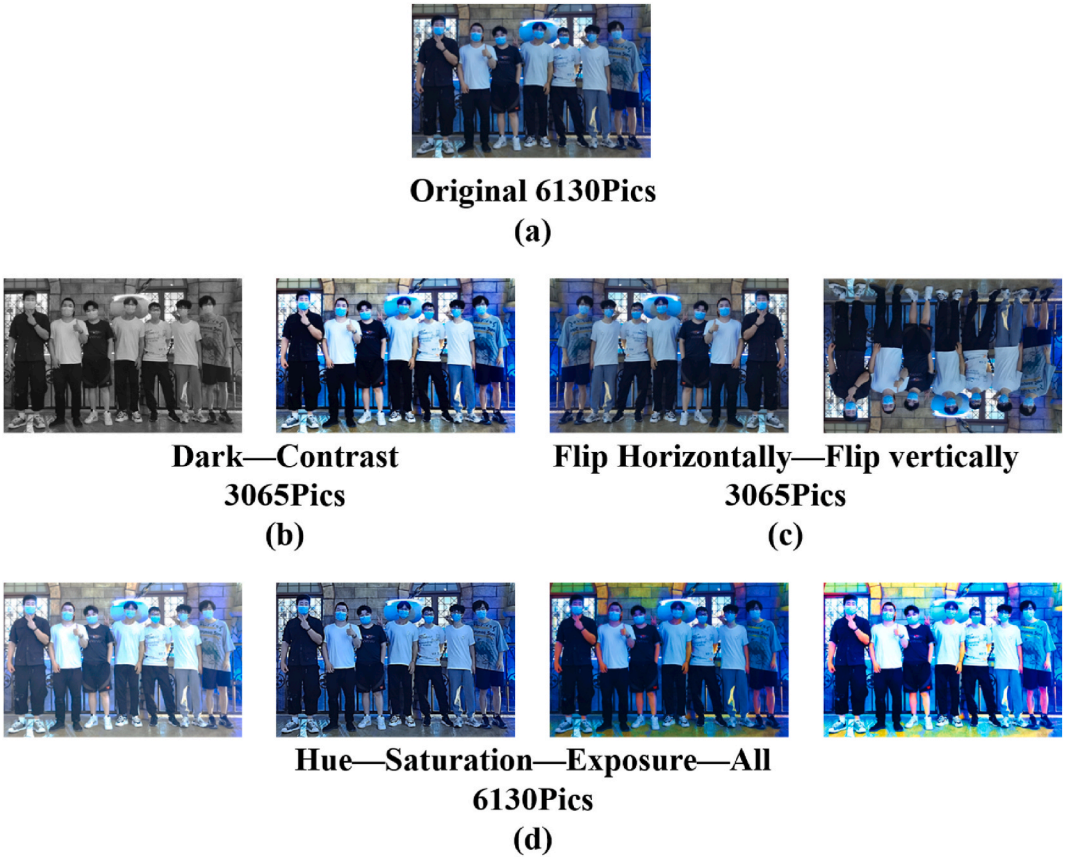



**Fig. 10.** Face mask dataset data augmentation. Data augmentation was performed on the (a) original dataset, through (b) dark-contrast, (c) horizontal-vertical flipping, and (d) hue-saturation-exposure adjustment, resulting in 3065, 3065, and 6130 images respectively. The augmented FMD dataset contains a total of 18,390 images.

**Table 2**
Experimental environment configuration.

| | | |
|---|---|---|
| **Hardware environment** | CPU | AMD Ryzen 7 5700X |
| | RAM<br>Video memory | 64 GB<br>12 GB |
| | GPU | NVIDIA GeForce RTX 3080Ti |
| **Software environment** | OS | Ubuntu |
| | CUDA Toolkit V11.4;<br>CUDNN V8.0.4;<br>Python 3.9.2;<br>torch 1.8.1; | |
| | torchvision 0.9.1; | |
| **Encode environment** | VSCode | |

$$R_c = \frac{TP_c}{TP_c + FN_c} \tag{14}$$

$$AP = \int P_c(R_c)d(R_c) \tag{15}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{16}$$

**Table 3**
Model experimental parameter settings.

| Parameter | Settings | Parameter | Settings |
|---|---|---|---|
| Seed | none | Warmup epochs | 5 |
| Num of class | 3 | Max epoch | 100 |
| Input size | (640,640) | Warmup learn rate | 0 |
| Degrees | 10.0 | No augmentation epochs | 15 |
| Translate | 0.1 | Min learn ratio | 0.05 |
| Scale | (0.1, 2) | Weight decay | 0.0005 |
| Mosaic | (0.8, 1.6) | Momentum | 0.9 |
| Shear | 2.0 | Test size | (640,640) |
| Enable mix-up | True | NMS threshold | 0.65 |

where $TP_c$ represents the true positives, $TN_c$ represents the true negatives, $FP_c$ represents the false positives and $FN_c$ represents the false negatives. AP measures the detection performance of a model for each class, while mAP is the average value of AP across all classes and is the primary metric for evaluating the overall performance of the model.

### 4.4. IYOLO-NL ablation experiments

To evaluate the impact of each component on the overall performance of IYOLO-NL, a series of ablation experiments were conducted. As illustrated in Table 4, the new sample assignment scheme (decoupled head, anchor-free, multi-level prediction, global dynamic-k label assignment strategy) improves the AP by 6.8% when compared with the baseline detector-YOLOv5l, among these components, the attention mechanism demonstrates the most significant impact on AP, with an increase of 2.7%. Despite a slight increase in the number of parameters, the IYOLO-NL detector incorporates lightweight structures such as SSPP and CSPNet-Ghost bottleneck to achieve an average FPS of 97.1, enabling it to perform localization inference on the object in a more efficient manner.

The decoupled head, which is closely associated with key components such as the anchor-free method and global dynamic-k label assignment strategy, is the central element of the IYOLO-NL detector. On the same backbone and neck structure, Fig. 11 (a) shows that the IYOLO-NL detector with the decoupled head achieved an AP that was approximately 15% higher in detecting small objects than that of the IYOLO-NL detector with the coupled head. Additionally, the IYOLO-NL detector with the decoupled head maintained high accuracy in detecting medium- and large-scale objects (Fig. 11 (b) and (c)).

To ensure objects remain within the receptive field of the feature maps, the maximum distance of the anchor points is limited. Furthermore, objects of varying sizes are distributed across different feature maps, with overlapping primarily occurring between objects that differ significantly in size [1,28,37,39]. Therefore, the multi-level prediction strategy with PANet-SC improves the detection performance of small and overlapping objects.

The impact of using the ghost module and SSPP on memory usage and training acceleration were analyzed. The ghost module includes an identity mapping and $\frac{n}{r}\cdot(r-1)$ linear operations, each with an average kernel size of $d \times d$ (e.g., $3 \times 3$ or $5 \times 5$). Compared to conventional convolutional operations ($n \cdot h \cdot w \cdot c \cdot k \cdot k$), the ghost module has a theoretical acceleration ratio ($r_a$) Eqs. (17) and (18):

$$r_a = \frac{n \cdot h \cdot w \cdot c \cdot k \cdot k}{\frac{n}{r}\cdot h \cdot w \cdot c \cdot k \cdot k + (r-1)\cdot\frac{n}{r}\cdot h \cdot w \cdot d \cdot d} \tag{17}$$

$$= \frac{c \cdot k \cdot k}{\frac{1}{r}\cdot c \cdot k \cdot k + \frac{r-1}{r}\cdot d \cdot d} \approx \frac{r \cdot c}{c + r - 1} \approx r \tag{18}$$

which is equal to its parameter compression ratio ($r_c$), the ghost module located in the IYOLO-NL backbone achieves r times higher inference speed and spatial computing efficiency compared to the original convolutional network Eq. (19).

**Table 4**
Results of ablation experiments on IYOLO-NL. Take YOLOv5l as a baseline, iteratively improved it based on the contents in methods column. The green numbers in parentheses in the AP column represent the improvement of the current version over the previous one. The FPS results are the average of multiple tests.

| Methods | AP(%) | Parameters | GFLOPs | Latency | FPS(Avg) |
|---|---|---|---|---|---|
| **YOLOv5l-Baseline** | 85.5 | 63.00 M | 157.3 | 10.5 ms | 95.2 |
| +Decoupled Head | 86.6(+1.1) | 63.86 M | 186.1 | 11.6 ms | 86.2 |
| +Strong Augmentation | 89.0(+2.4) | 63.86 M | 186.1 | 11.6 ms | 86.2 |
| +Anchor-Free | 90.9(+1.9) | 63.71 M | 185.3 | 10.2 ms | 90.1 |
| +Multi-Positives | 92.6(+1.7) | 63.71 M | 185.3 | 10.2 ms | 90.1 |
| +Global Dynamic-K | 94.7(+2.1) | 63.71 M | 185.3 | 10.2 ms | 93.1 |
| +Attention Mechanism | 97.4(+2.7) | 63.75 M | 191.6 | 10.4 ms | 96.6 |
| **IYOLO-NL** | **98.7**(+12.5) | 63.76 M | 190.7 | 10.4 ms | 97.1 |

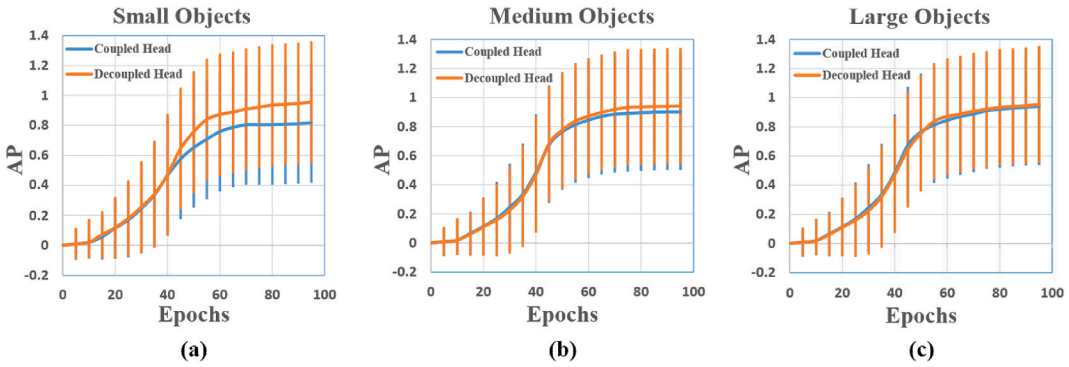



**Fig. 11.** Multi-scale object detection performance of IYOLO-NL with various heads. The sky-blue solid line represents the IYOLO-NL model with the original YOLOv5 detection head, while the orange solid line represents the IYOLO-NL model with the decoupled head. Both models were evaluated on (a) small, (b) medium, and (c) large objects, with the error bars displaying the average error across multiple 10-fold cross-validations.

$$r_c = \frac{n \cdot c \cdot k \cdot k}{\frac{n}{r} \cdot c \cdot k \cdot k + (r-1) \cdot \frac{n}{r} \cdot d \cdot d} \approx \frac{r \cdot c}{c + r - 1} \approx r \tag{19}$$

In contrast to the SPP bottleneck [47], the IYOLO-NL detector achieves equivalent detection results using the SSPP bottleneck, which exhibits superior training efficiency and remarkable progress, especially in forward and backward calculations. Table 5 provides additional information on SSP and SSPP, demonstrating that the SSPP bottleneck significantly reduces processing time and saves GPU memory usage.

### 4.5. Compare experiments with SOTA object detectors

Fig. 12 and Table 6 illustrate the performance of various state-of-the-art (SOTA) object detectors on FMD. Each model was evaluated separately on the large, medium, and small datasets of the FMD dataset using the data partitioning method described in subsection 4.1. With the exception of Faster *R*–CNN [2], object detection models tend to shift towards the lower-right corner as the size of the target object decreases, indicating that detection accuracy and latency are highest for large objects and lowest for small objects. However, the frame rate (FPS) is higher for smaller objects.

Under the same experimental environment and training strategies, IYOLO-NL achieved the best performance with an accuracy of 98.8%, a best AP of 98.7%, a mAP of 95.7%, and a best FPS of 130. It is noteworthy that IYOLO-NL outperformed Faster *R*–CNN in FPS by almost 22 times. Compared to other single-stage detectors, such as SSD [20], EfficientDet [54], benchmark-YOLOv5l, and novel YOLOv6∼YOLOv8 [3,29], the IYOLO-NL model is located at the top right corner, indicating that it achieved the high AP and mAP while maintaining highest FPS, almost 4 times faster than the slowest.

### 4.6. IYOLO-NL none left performance compared to baseline YOLOv5l

IYOLO-NL exhibits substantial improvement over YOLOv5l in terms of evaluation metrics. Fig. 13 (a) and (b) illustrate that IYOLO-NL outperforms YOLOv5l in the early stages and consistently maintains superior performance, with higher AP and mAP values. Additionally, IYOLO-NL exhibits robustness and stability in real-time mask-wearing detection when compared to the baseline YOLOv5l.

Firstly, as shown in Fig. 14 (a) and (c), when dealing with complex backgrounds, YOLOv5l exhibited "False Detection" phenomena where the background was mistakenly detected as a positive object. In contrast, IYOLO-NL demonstrated robustness in handling complex backgrounds, as evidenced by its superior performance in Fig. 14 (b) and (d) in the comparative results.

Secondly, when dealing with dense scenes containing multiple-scale objects, YOLOv5l exhibited significant instances of false negatives in its detection results for Fig. 15 (a), (b), (e), and (f), implying YOLOv5l missed some positive objects. In contrast, IYOLO-NL accurately detected multiple objects in such crowded scenarios, as demonstrated by its corresponding results for Fig. 15 (c), (d), (g),

**Table 5**
Performance comparison of SPP and SSPP structures. Comparison experiments were conducted based on the same IYOLO-NL architecture. The compared parameters include parameter size (Params, unit: memory size), billion floating-point operations per second (GFLOPS), GPU usage (unit: GB), forward propagation time (Forward, unit: milliseconds), backward propagation time (Backward, unit: milliseconds), and input-output feature size (I/O).

| Bottleneck | Params | GFLOPS | GPU | Forward | Backward | I/O |
|---|---|---|---|---|---|---|
| SPP | 2.6 M | 334 | 4.586 GB | 80.90 ms | 171.2 ms | (16,20,20,1024) |
| SSPP | 2.6 M | 334 | 4.452 GB | 55.23 ms | 114.9 ms | (16,20,20,1024) |

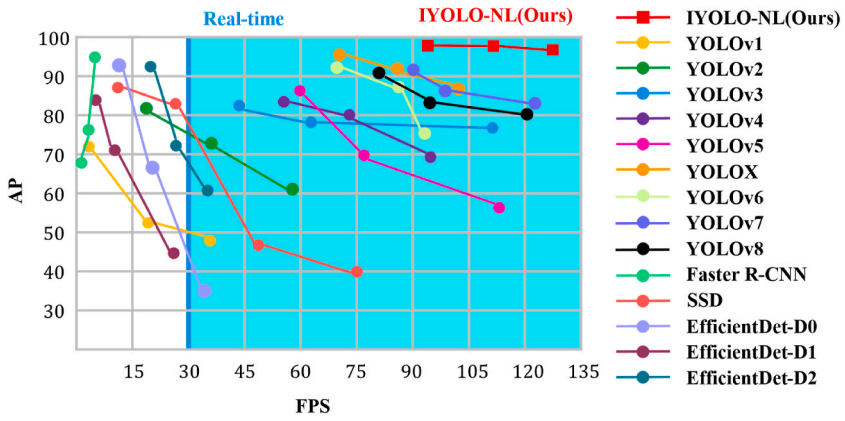



**Fig. 12.** The performances of IYOLO-NL and other SOTA object detectors. Deployed 15 object detection models to test the FMD dataset. Each model corresponds to a scatter plot, where the scatter points from left represent the best AP and corresponding FPS for the three label categories of large-, medium- and small-size data, respectively. The trend of the Faster *R*–CNN model is the opposite.

**Table 6**
The performances of IYOLO-NL and other SOTA object detectors. Compare in input size, Accuracy, AP, mAP and FPS. The indicators are taken from the best performance of each model. All YOLO series detectors (l-type) and other state-of-the-art (SOTA) models were trained and evaluated using the same dataset and experimental settings as IYOLO-NL.

| Model | Input Size | Accuracy(%) | AP(%) | mAP(%) | FPS |
|---|---|---|---|---|---|
| YOLOv1 [21] | 448 × 448 | 71.3% | 66.5% | 61.2% | 35 |
| YOLOv2 [22] | 448 × 448 | 80.8% | 81.2% | 79.3% | 54 |
| YOLOv3 [7] | 416 × 416 | 89.7% | 82.2% | 81.5% | 110 |
| YOLOv4 [1] | 416 × 416 | 93.2% | 83.1% | 80.7% | 95 |
| YOLOv5 | 640 × 640 | 93.4% | 85.5% | 82.8% | 115 |
| YOLOX [28] | 640 × 640 | 97.3% | 96.1% | 93.7% | 103 |
| YOLOv6 [29] | 640 × 640 | 96.5% | 91.7% | 91.1% | 97 |
| YOLOv7 [3] | 640 × 640 | 95.4% | 93.6% | 92.4% | 124 |
| YOLOv8 | 640 × 640 | 94.9% | 91.1% | 90.6% | 121 |
| Faster *R*–CNN [2] | 600 × 600 | 94.4% | 91.3% | 92.1% | 6 |
| SSD [20] | 300 × 300 | 89.8% | 87.6% | 84.2% | 75 |
| EfficientDet-D0 [54] | 512 × 512 | 91.5% | 90.2% | 87.6% | 32 |
| EfficientDet-D1 [54] | 640 × 640 | 90.7% | 85.6% | 83.4% | 28 |
| EfficientDet-D2 [54] | 736 × 736 | 93.2% | 91.6% | 89.9% | 33 |
| **IYOLO-NL** | **640 × 640** | **98.8%** | **98.7%** | **95.7%** | **130** |

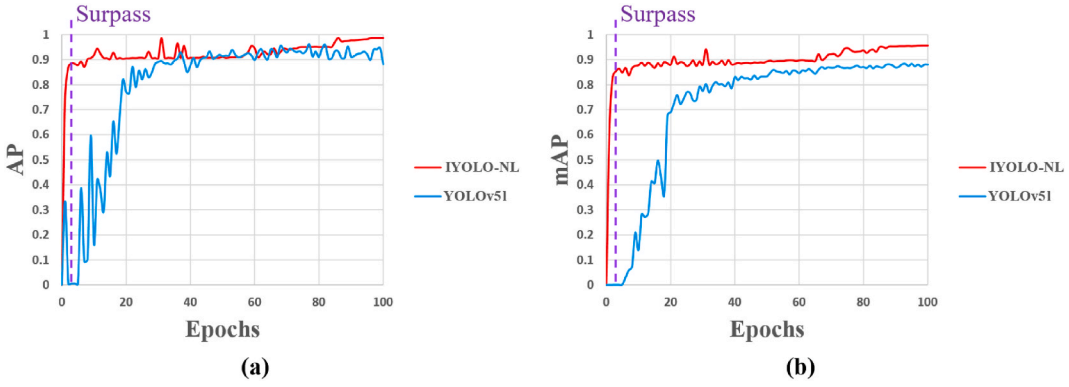



**Fig. 13.** The performance comparison between IYOLO-NL and YOLOv5l. The red solid line represents IYOLO-NL, and the blue solid line represents YOLOv5l. Both models were validated on the FMD dataset with 100 epochs as the benchmark (x-axis), and the y-axis represents: (a) AP, (b) mAP. The purple vertical dashed lines in figures indicate the position where IYOLO-NL first surpassed YOLOv5l in each evaluation metric.

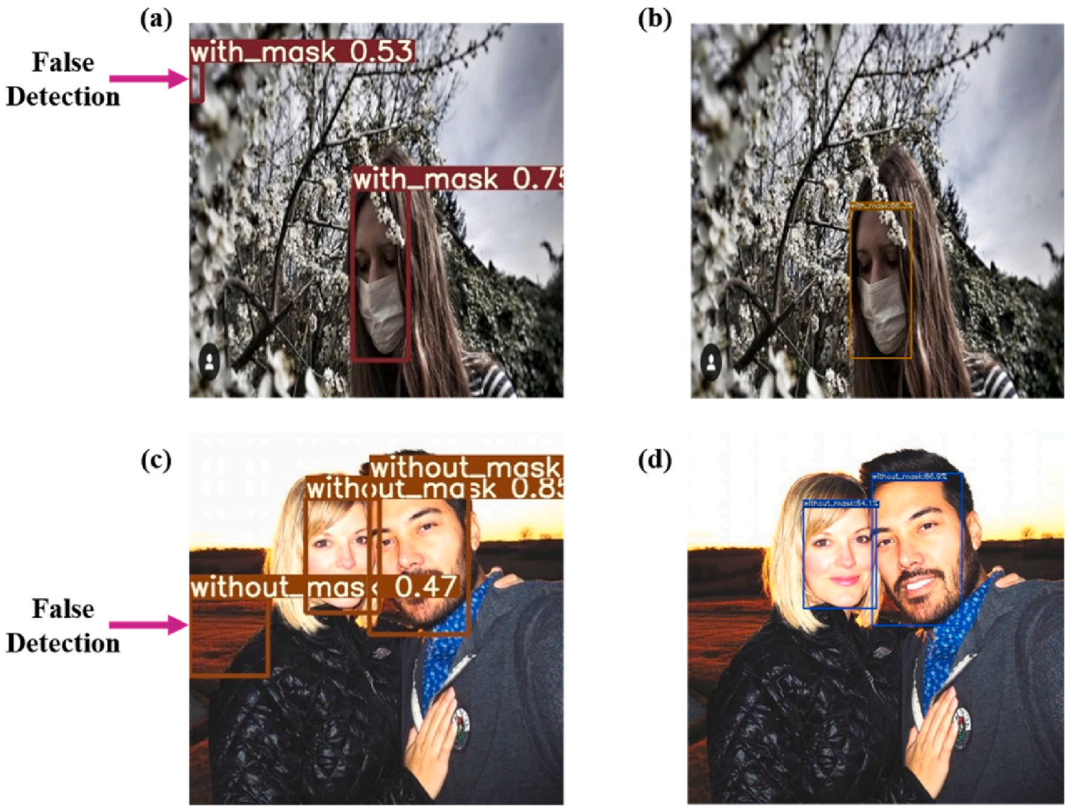



**Fig. 14.** Comparison of the detection effects of YOLOv5l and IYOLO-NL. The detection results of YOLOv5l are (a), (c), while the corresponding results of IYOLO-NL are (b), (d).

and (h).

These findings highlight the advantages of IYOLO-NL over YOLOv5l in challenging scenarios with complex and dense backgrounds and suggest its potential for practical applications in real-world settings.

### *4.7. IYOLO-NL performance on FMD*

Fig. 16 demonstrates the excellent detection performance of IYOLO-NL on the FMD dataset, particularly in handling dense small object scenes. IYOLO-NL can detect small faces in walls or albums without any errors, including small headshots on healthcare worker's ID cards (Fig. 16 (a) and (b)). Even when the nose and mouth area is obstructed by objects such as arms, backs of hands, and teacups, IYOLO-NL can correctly identify and classify them without any omissions (Fig. 16 (c)). IYOLO-NL can successfully detect three different categories of mask-wearing situations, including multi-scale, overlapping, and small-occluded objects in dense scenes. Additionally, IYOLO-NL exhibits style transfer ability, as illustrated in the last two pictures in Fig. 16 (c), where even puppets with face masks are correctly detected.

### *4.8. Comparison of IYOLO-NL and previous studies*

We compared the detection performance of IYOLO-NL with other methods, including traditional handcrafted methods and various-stage neural network methods trained on different sources (i.e., datasets, environments) and evaluated with different indicators, as shown in Table 7. Despite the varying comparative conditions, IYOLO-NL demonstrates the best detection performance among all the methods, especially in FPS, by comprehensively processing three categories.

### *4.9. Potential and limitations of the IYOLO-NL*

Our proposed IYOLO-NL model is versatile and can be applied to any real-world object detection task. Although we only demonstrate its application to face mask detection in this paper, it covers more than three types of objects: faces, masks, hands, clothing, and any other objects that may occlude the face area in real-world scenarios. The comprehensive performance of our model far exceeds that of previous research methods. Specifically, IYOLO-NL achieves a score of 99.6% AP in face detection, which is a well-established research area.

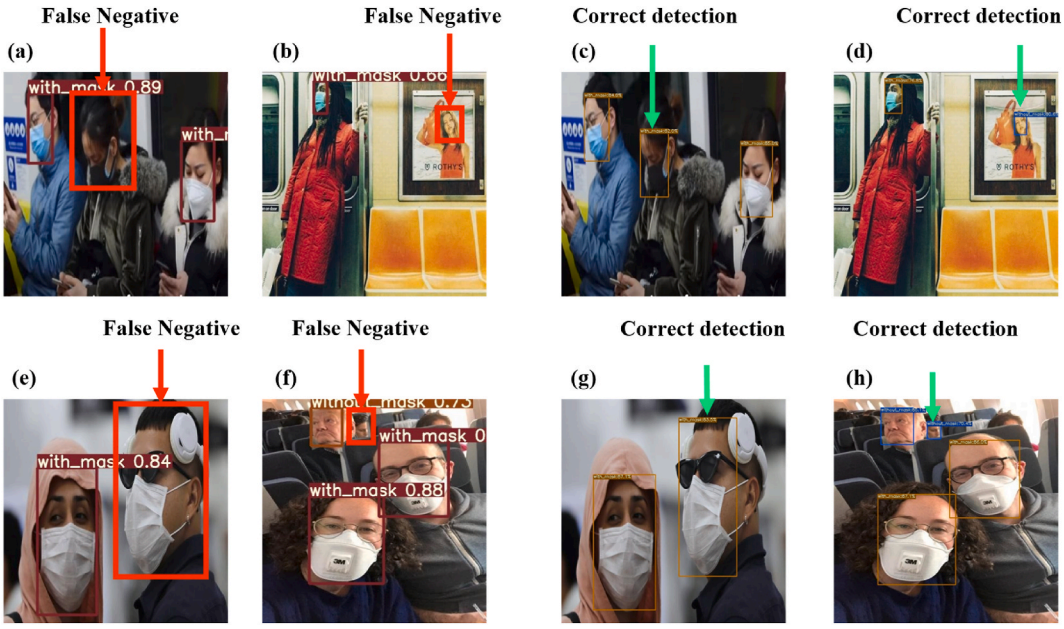



**Fig. 15.** Comparison of the detection effects of YOLOv5l and IYOLO-NL on crowded objects. The detection results of YOLOv5l are (a), (b), (e), and (f), while the corresponding results of IYOLO-NL are (c), (d), (g), and (h).

We categorized the dataset into three groups based on size: large, medium, and small. Despite the conventional limitations of the YOLO detector, our study found that novel IYOLO-NL model performs better than the YOLOv5 baseline in detecting multi-scale objects. The detection of puppets in our results confirmed the transferability of IYOLO-NL. Furthermore, IYOLO-NL's anchor-free design eliminates the need for parameter settings and enables effortless switching between datasets.

In practical applications, IYOLO-NL achieves an average of 97 FPS with 98.8% accuracy during real-world inference, fulfilling the requirement for high-precision and real-time monitoring. The proposed IYOLO-NL model has the potential to enhance safety, reduce costs, and improve efficiency in various real-world applications, such as public safety, transportation, and industrial automation.

IYOLO-NL is a single-stage object detection model optimized for GPU training and inference, which can efficiently utilize GPU resources compared to the current mainstream YOLO models. However, its performance on edge devices (such as smartphones and microcontrollers that rely on CPU for computation) may be limited. To enhance its efficacy on devices with limited GPU resources, we suggest adjusting the dynamic-k parameter of the global dynamic label assignment strategy or considering using a MobileNet structure instead of the IYOLO-NL backbone.

## 5. Conclusion

In this paper, a novel real-time object detector IYOLO-NL was proposed. IYOLO-NL redefined the manner of sample assignment by using novel global dynamic-k label assignment strategy in an anchor-free fashion. To reduce computational complexity and enhance inference speed, the developed CSPNet-Ghost bottleneck and SSPP network were utilized in the backbone. In the Neck part, the IYOLO-NL employed the proposed PANet-SC with multi-level prediction scheme to cope with multi-scale, overlapping and small objects more effectively and accurately. To avoid misalignment problems and improve prediction performance, IYOLO-NL adopted a decouple head as the model output medium.

The FMD dataset and IYOLO-NL model were constructed for face mask detection. Several experiments indicated that IYOLO-NL outperforms the baseline YOLOv5l and other methods with 98.8% accuracy, 95.7% mAP and 130 FPS. IYOLO-NL demonstrated outstanding performance in addressing complex backgrounds, overlapping objects, and other related challenges, highlighting its robustness and "None Left" characteristic. IYOLO-NL takes a leading position in the real-time face mask detection field when compared with SOTA detectors such as YOLOX, YOLOv6, YOLOv7 and YOLOv8. Along with the improvement of the YOLO models, we hope IYOLO-NL can achieve better performance in the future.

## Ethics statement

Due to the characteristics of the study, it does not require approval by the ethics committee. All participants provided informed consent for the publication of their images.


**Fig. 16.** Detection results of IYOLO-NL on FMD dataset. (a) Three label detection results. (b) Dense and Small objects detection results. (c) Overlaying, occlude and style transfer objects detection results.

**Table 7**
Comparison of IYOLO-NL and previous studies. The highlighted blue entries are our IYOLO-NL. The comparison range covers conventional, multi-stage, two-stage, and single-stage algorithms. The classes list indicates the number of classifications for the tested objects (correctly wearing masks, not wearing masks, not correctly wearing masks, etc.). The datasets list shows the source of the original data. Some papers use accuracy and precision interchangeably (or separately), so the highest numerical value of two indicators in that column is displayed. The data marked "-" in the table indicates that there is no corresponding description in the corresponding paper.

| Category | Method | Classes | Datasets | Accuracy&Precious | mAP | FPS |
|---|---|---|---|---|---|---|
| Conventional | Dewantara et al. [12] | 2 | 1000 images, self-built | 86.9% | – | 25 |
| | Nieto-Rodriguez et al. [55] | 2 | 677 test cases, self-built | 95% | – | 10 |
| | Petrovic et al. [56] | 3 | – | 84%–91% | – | 38 |
| | Fang et al. [57] | 2 | 6024 images, self-built | 96.5% | | 46 |
| Multi-Stage | Cota et al. [14] | 2 | 2270 images, self-built | – | 85.92% | 15 |
| | Lin et al. [15] | 2 | 992 images, self-built | Daytime:95.8%<br>Nighttime:94.6% | – | – |
| | Qin et al. [58] | 3 | 3835 images, self-built | 98.7% | – | 33 |
| Two-Stage | Mercaldo et al. [16] | 2 | 4095 images | 98% | – | – |
| | Zereen et al. [17] | 2 | 5504 images, self-built | 97.13% | – | – |
| | Rudraraju et al. [59] | 3 | 1270 images, self-built | 90% | – | – |
| Single-Stage | Zhang et al. [60] | 3 | 4672 images, self-built | – | 84.1% | – |
| | Deng et al. [61] | 2 | 3656 images, self-built | – | 91.7% | – |
| | Loey et al. [23] | 1 | 1415 images, Kaggle | – | 81% | – |
| | Jiang et al. [25] | 3 | 9205 images, self-built | – | 73.7% | 16 |
| | Sharma [62] | 2 | – | – | 60% | – |
| **IYOLO-NL** | **Ours** | **3** | **6130 images, self-built** | **98.8%** | **95.7%** | **130** |

## Author contribution statement

**Yan Zhou:** Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper.

## Data availability statement

Data will be made available on request.

## Consent for publication

All authors have given consent for publication.

## Funding

This research received no funding.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

## Acknowledgement

We thank Ultralytics for open sourcing the YOLOv5 and YOLOv8 codes.

## Appendix

**Table A1**
Small Sieve pseudocode.

**Algorithm:** **Small Sieve**

**Input:**
**Cost:** The cost matrix
**M:** The matching matrix
**IM:** The IoU matrix
**$B_{gtbb}$:** The ground-truth bounding box
**$B_{pred}$:** The candidate prediction box/candidate anchor points
**AMG: T**he anchor point matching with ground-truth bounding box matrix
**MM: T**he mask matrix

**Output:**
**num:** Number of candidate anchor points
**ATG**: The anchor point to one ground-truth bounding box matrix
**MI**: The matched IoU matrix

**Begin**:

**1:** $\text{Cost} \leftarrow c_{ij} = L_{ij}^{cls} + \lambda L_{ij}^{reg}$
**2:** $M_{Cost}^{size} \leftarrow 0$
**3:** $\text{IM} \leftarrow \text{pairwise iou}(B_{gtbb}, B_{pred})$
**4:** Candidate anchor points for one $B_{gtbb}$: $n \leftarrow \min\{10, \text{IM}\}$
**5:** Pick anchor points: $\text{topk\_ious} \leftarrow \text{topk}(\text{IM}, n)$
**6:** Calculate k for $B_{gtbb}$: $\text{dynamic} - k \leftarrow \min(\lfloor \sum \text{topk\_ious} \rfloor, 1)$
**7:** **for** ij in $B_{pred}$ do BFS
**8:** $B \leftarrow \text{BFS}(\text{Cost})$
**9:** Find the mini k location mask: $\text{pidx} \leftarrow \min(B, \text{dynamic} - k)$
**10:** The matching anchor points: $M(\text{pidx}) \leftarrow 1$
**11:** **end for**
**12:** Sum by row: $\text{AMG} = \sum_{\text{row}} M$
**13:** **if** $\text{AMG} > 1$
**14:** $D \leftarrow \text{Dijkstra}(\text{Cost}, \text{AMG} > 1)$
**15:** $M(\text{AMG} > 1) \leftarrow 0$
**16:** The mini cost vector marked as one: $M(D) \leftarrow 1$
**17:** **end if**
**18:** Find the column which >0: $\text{MM} \leftarrow \sum_{\text{column}} M > 0$
**19:** The total number of candidate anchor points: $\text{num} \leftarrow \sum \text{MM}$
**20:** Let one anchor point to one $B_{gtbb}$: $\text{ATG} \leftarrow \text{argmax}\{M(\text{MM})\}$
**21:** The classes of matched anchor points: $B_{pred}(\text{class}) \leftarrow B_{gtbb}(\text{class})$
**22:** The matched IoU: $\text{MI} \leftarrow \sum_{\text{MM}} \left(M \times \text{pairwise iou}(B_{pred}, B_{gtbb})\right)$
**23:** **return** $\text{num}, B_{pred}(\text{class}), \text{ATG}, \text{MI}$

**End**

**Step 1**. Construct the cost matrix from the loss function.

**Step 2**. Initialize the matching matrix between the ground-truth bounding box and the initial candidate points and assign all values in the matching matrix to 0, indicating that they are not matched.

**Step 3**. Calculate the IoU between the ground-truth bounding boxes and the candidate prediction boxes. Find the top-10 largest IoUs and their corresponding data in the IoU matrix.

**Step 4**. Sum up the IoUs of the top-10 samples, then count the number of anchor points allocated to the current ground-truth bounding box, i.e., the dynamic-k.

**Step 5**. For each ground-truth bounding box, use the breadth-first search algorithm (BFS) to calculate the positions of the k anchor points with the smallest cost values.

**Step 6**. Filter out the shared candidate anchor points, i.e., eliminate the cases where one candidate anchor point is matched to

multiple ground-truth bounding boxes. To be specific, calculate the position with the smallest loss value through the Dijkstra algorithm among the shared candidate anchor points, set the mask of that position to 1, and set the rest of the positions in the same column to 0.

**Step 7**. Return the number of candidate anchor points, the matching information (IoU value, ground-truth bounding box index, etc.) between anchor points and ground-truth bounding boxes, and update the positive sample mask.

## References


[1] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: Optimal Speed and Accuracy of Object Detection, 2020 arXiv preprint arXiv:2004.10934.
[2] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, 2015.
[3] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, 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, 2023.
[4] A. Cabani, et al., MaskedFace-Net–A dataset of correctly/incorrectly masked face images in the context of COVID-19, Smart Health 19 (2021), 100144.
[5] N. Faruqui, et al., Trackez: an IoT-Based 3D-Object Tracking from 2D Pixel Matrix Using Mez and FSL Algorithm, IEEE Access, 2023.
[6] N. Faruqui, M.A. Yousuf, Performance-accuracy optimization of face detection in human machine interaction, in: 2019 5th International Conference on Advances in Electrical Engineering (ICAEE), IEEE, 2019.
[7] J. Redmon, A. Farhadi, Yolov3: an Incremental Improvement, 2018 arXiv preprint arXiv:1804.02767.
[8] X. Zhao, et al., A single stream network for robust and real-time RGB-D salient object detection, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16, Springer, 2020.
[9] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR, 2001 Ieee, 2001.
[10] T. Ojala, M. Pietikäinen, D. Harwood, A comparative study of texture measures with classification based on featured distributions, Pattern Recogn. 29 (1) (1996) 51–59.
[11] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005, Ieee, 2005.
[12] B.S.B. Dewantara, D.T. Rhamadhaningrum, Detecting multi-pose masked face using adaptive boosting and cascade classifier, in: 2020 International Electronics Symposium (IES), IEEE, 2020.
[13] T. He, Mask wearing detection method based on the skin color and eyes detection, in: Journal of Physics: Conference Series, IOP Publishing, 2021.
[14] D.A.M. Cota, Monitoring COVID-19 Prevention Measures on CCTV Cameras Using Deep Learning, Politecnico di Torino, 2020.
[15] H. Lin, et al., Near-realtime face mask wearing recognition based on deep learning, in: 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC), IEEE, 2021.
[16] F. Mercaldo, A. Santone, Transfer learning for mobile real-time face mask detection and localization, J. Am. Med. Inf. Assoc. 28 (7) (2021) 1548–1554.
[17] A.N. Zereen, et al., Two-stage facial mask detection model for indoor environments, in: Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020, Springer, 2021.
[18] T.Q. Vinh, N.T.N. Anh, Real-time face mask detector using YOLOv3 algorithm and Haar cascade classifier, in: 2020 International Conference on Advanced Computing and Applications (ACOMP), IEEE, 2020.
[19] S. Lin, et al., Masked face detection via a modified LeNet, Neurocomputing 218 (2016) 197–202.
[20] W. Liu, et al., Ssd: single shot multibox detector, in: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–*14, 2016, Proceedings*, Springer, 2016.
[21] J. Redmon, et al., You only look once: unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
[22] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[23] M. Loey, et al., Fighting against COVID-19: a novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection, Sustain. Cities Soc. 65 (2021), 102600.
[24] M.R. Bhuiyan, S.A. Khushbu, M.S. Islam, A deep learning based assistive system to classify COVID-19 face mask for human safety with YOLOv3, in: 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2020.
[25] X. Jiang, et al., Real-time face mask detection method based on YOLOv3, Electronics 10 (7) (2021) 837.
[26] J. Yu, W. Zhang, Face mask wearing detection algorithm based on improved YOLO-v4, Sensors 21 (9) (2021) 3263.
[27] G. Yang, et al., Face mask recognition system with YOLOV5 based on image recognition, in: 2020 IEEE 6th International Conference on Computer and Communications (ICCC), IEEE, 2020.
[28] Z. Ge, et al., Yolox: Exceeding Yolo Series in 2021, 2021 arXiv preprint arXiv:2107.08430.
[29] C. Li, et al., YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications, 2022 arXiv preprint arXiv:2209.02976.
[30] C.-Y. Wang, et al., 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, 2020.
[31] G. Song, Y. Liu, X. Wang, Revisiting the sibling head in object detector, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.
[32] H. Li, et al., Pruning Filters for Efficient Convnets, 2016 arXiv preprint arXiv:1608.08710.
[33] M. Rastegari, et al., Xnor-net: imagenet classification using binary convolutional neural networks, in: European Conference on Computer Vision, Springer International Publishing, Cham, 2016.
[34] A.G. Howard, et al., Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017 arXiv preprint arXiv:1704.04861.
[35] G. Hinton, O. Vinyals, J. Dean, Distilling the Knowledge in a Neural Network, 2015 arXiv preprint arXiv:1503.02531.
[36] C. Li, et al., Yolov6 V3. 0: A Full-Scale Reloading, 2023 arXiv preprint arXiv:2301.05586.
[37] Z. Tian, et al., Fcos: fully convolutional one-stage object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019.
[38] X. Zhou, D. Wang, P. Krähenbühl, Objects as points, arXiv preprint arXiv (2019), 1904.07850.
[39] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, Scaled-yolov4: scaling cross stage partial network, in: Proceedings of the IEEE/cvf Conference on Computer Vision and Pattern Recognition, 2021.
[40] T.-Y. Lin, et al., Feature pyramid networks for object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[41] S. Liu, et al., Path aggregation network for instance segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[42] C. Feng, et al., Tood: task-aligned one-stage object detection, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE Computer Society, 2021.
[43] Z. Ge, et al., Optimal transport assignment for object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.
[44] S. Zhang, et al., Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.
[45] S. Lin, et al., Local patch autoaugment with multi-agent collaboration, IEEE Trans. Multimed. (2023).
[46] B. Zhu, et al., Autoassign: Differentiable Label Assignment for Dense Object Detection, 2020 arXiv preprint arXiv:2007.03496.

[47] K. He, et al., Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE Trans. Pattern Anal. Mach. Intell. 37 (9) (2015) 1904–1916.
[48] K. Han, et al., Ghostnet: more features from cheap operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.
[49] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[50] K. He, et al., Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
[51] S. Woo, et al., Cbam: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision, ECCV), 2018.
[52] A. Vaswani, et al., Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[53] H. Zhang, et al., mixup: beyond empirical risk minimization, arXiv preprint arXiv (2017), 1710.09412.
[54] M. Tan, R. Pang, Q.V. Le, Efficientdet: scalable and efficient object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.
[55] A. Nieto-Rodriguez, M. Mucientes, V.M. Brea, System for medical mask detection in the operating room through facial attributes, June 17-19, 2015, in: Pattern Recognition and Image Analysis: 7th Iberian Conference, IbPRIA 2015 vol. 7, Proceedings, Santiago de Compostela, Spain, 2015 (Springer).
[56] N. Petrović, Đ. Kocić, IoT-based System for COVID-19 Indoor Safety Monitoring, IcETRAN Belgrade, 2020.
[57] T. Fang, X. Huang, J. Saniie, Design flow for real-time face mask detection using PYNQ system-on-chip platform, in: IEEE International Conference on Electro Information Technology (EIT). 2021, IEEE, 2021.
[58] B. Qin, D. Li, Identifying facemask-wearing condition using image super-resolution with classification network to prevent COVID-19, Sensors 20 (18) (2020) 5236.
[59] S.R. Rudraraju, N.K. Suryadevara, A. Negi, Face mask detection at the fog computing gateway, in: 2020 15th Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2020.
[60] J. Zhang, et al., A novel detection framework about conditions of wearing face mask for helping control the spread of COVID-19, IEEE Access 9 (2021) 42975–42984.
[61] H. Deng, et al., Improved mask wearing detection algorithm for SSD, in: Journal of Physics: Conference Series, IOP Publishing, 2021.
[62] V. Sharma, Face Mask Detection Using Yolov5 for COVID-19, California State University San Marcos, 2020.