A Method of Pneumonia Detection Based on an Improved YOLOv5s

Ruiqing Shan, Xiaoxia Zhang, Shicheng Li

ABSTRACT—The advancement of pneumonia detection technology is crucial for the automation and the enhancement of diagnostic accuracy in pneumonia. Recent developments have seen the application of automatic detection methods utilizing deep learning in medical imaging. However, the diagnosis of pneumonia is often hampered by the small size and indistinct boundaries of lesions. To enhance diagnostic efficiency and mitigate errors due to human factors, this paper introduces an improved pneumonia detection algorithm, YOLOv5-TAF. First, the incorporation of the Triplet Attention module into the backbone network establishes inter-dimension dependencies, augmenting model accuracy. Second, Adaptive Spatial Feature Fusion (ASFF) is employed to dynamically learn and apply spatial weights to each scale feature map, effectively filtering out conflicting information and enhancing the network's capability to integrate features of varying scales. Additionally, the substitution of the original activation function with FReLU increases the sensitivity of the activation space and markedly improves image quality. Comprehensive experimental analyses on the Chest X-Ray Images dataset have demonstrated that the YOLOv5-TAF algorithm achieves a mean Average Precision (mAP) of 91.71%, which is 3.57% higher than the YOLOv5s algorithm, with a detection speed of 58.72 frames/s. When compared with faster-RCNN, SSD, YOLOv8s, and other object detection algorithms, YOLOv5-TAF shows superior practicability in pneumonia detection.

***Index Terms*—Pneumonia; YOLOv5s; Multi-scale feature fusion; Attention mechanism**

I. Introduction

PNEUMONIA is one of the most prevalent infectious diseases encountered in clinical medicine. It is characterized by a sudden onset, rapid progression, and complex manifestations, posing a significant threat to human survival and health. The World Health Organization reports that pneumonia is responsible for the deaths of three to five million individuals annually. Despite being a common pulmonary ailment, achieving an accurate and swift diagnosis remains a formidable challenge.

Currently, the primary clinical diagnostic tools for pulmonary diseases include X-ray, computed tomography (CT), and magnetic resonance imaging (MRI), etc [1]. are often the preferred option for most patients due to their cost-effectiveness and reasonable imaging quality. However, X-rays suffer from a lack of color and texture information, and the quality of focused tissue imagery can be adversely affected by environmental factors, complicating the differentiation between affected and normal tissues. Consequently, diagnosing demands a high level of professional knowledge and clinical experience from physicians. Moreover, the process of examining and diagnosing lung X-rays is time-consuming and susceptible to errors, including misdiagnoses and omissions, often as a result of physician fatigue or the presence of other lung conditions.

In recent years, advancements in software and hardware technologies have facilitated the integration of deep learning techniques into various medical applications, notably in medical imaging. This includes the construction of images by imaging equipment, detection of chest X-rays, identification of breast diseases, diagnosis of brain conditions, and analysis of pathological specimens [2]. Utilizing deep learning for the diagnosis of pneumonia through chest X-ray images has proven to significantly enhance the ability to identify, classify, and quantify medical images. By analyzing a vast array of chest X-ray images from pneumonia patients, deep learning methods yield more precise outcomes compared to traditional approaches. Therefore, developing intelligent detection algorithms for pneumonia, utilizing deep learning, holds substantial social importance.

There are primarily two categories of target detection algorithms: two-stage detection and one-stage detection. The former involves generating candidate regions initially and then processing them through a deep network to extract features. Girshick et al. [3] introduced the use of convolutional neural networks (CNNs) over traditional methods like Scale-invariant Feature Transform (SIFT) [4] and Histogram of Oriented Gradients (HOG) [5] for feature extraction, employing Support Vector Machine (SVM) for classification to develop the Region Convolutional Neural Network (RCNN) detection algorithm. RCNN ensures stability and accuracy through the integration of CNNs, multiple layers of pooling, and activation functions. However, it relies on the Selective Search (SS) algorithm [6] to identify potential targets based on color and texture similarity and requires storing features extracted by CNNs on disk, which diminishes detection efficiency. Girshick et al. enhanced RCNN by introducing Fast-RCNN and Faster-RCNN [7]. Fast-RCNN improves detection efficiency [8] through the sharing of intermediate features, integrating the Region Proposal Network (RPN) with Fast-RCNN, eliminating the need for SS and the storage of intermediate features, thereby achieving speeds nearing real-time detection. In contrast, one-stage detection offers a more streamlined process. It employs a convolutional neural

Manuscript received September 18, 2023; revised May 10, 2024.

R. Q. Shan is a postgraduate student of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, 114051, China, e-mail: 1310846716@qq.com.

X. X. Zhang is a Professor of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, 114051, China (Corresponding author, phone: 86-0412-5929812; e-mail: aszhangxx@163.com).

S. C. Li is a postgraduate student of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, 114051, China, e-mail: 1056133593@qq.com.

network to directly extract features from the input image, culminating in output results based on regression. When compared to two-stage detection, one-stage methods have shown enhanced detection speed and incrementally improved accuracy. In 2016, Redmon et al. proposed the You Only Look Once (YOLO) algorithm, which foregoes the selection of regions, processes the entire image as input, and directly outputs the bounding box positions and categories at the output layer. Relative to Faster-RCNN, YOLO significantly simplifies the object detection process, markedly accelerating detection speed to support real-time applications [9].

Addressing the issue of misdiagnoses and missed diagnoses prevalent in existing pneumonia detection algorithms, this paper introduces the YOLOv5-TAF algorithm, an advancement on the YOLOv5s, designed to aid physicians in achieving rapid and precise diagnoses. Initially, while the YOLO series expedites the target detection process, it equates the contribution of the Intersection-over-Union (*IoU*) error across both large and small targets to the loss function, adversely affecting the algorithm's accuracy in localizing targets. To counter this, enhancements in detection accuracy are achieved by incorporating the Triplet Attention mechanism into the backbone network. Furthermore, this iteration improves upon the original YOLOv5s model's PANet feature fusion network with adaptively spatial feature fusion (ASFF), enhancing the network's ability to integrate features from targets of varying sizes. Additionally, replacing the YOLOv5s algorithm's original *SiLU* activation function with *FReLU* elevates the activation space's sensitivity and markedly enhances image clarity. Extensive testing on datasets has shown that the YOLOv5-TAF algorithm exceeds the traditional YOLOv5s in mean Average Precision (*mAP*) for pneumonia detection.

II. Materials And Method

A. YOLOv5 Object Detection Algorithm

YOLOv5 [10] is the latest installment in the YOLO series of target detection algorithms, marking a departure from its predecessors, YOLOv3 and YOLOv4, which featured only a lightweight tiny model alongside a full-sized model. YOLOv5 introduces four distinct network models: YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x [11]. The architecture's width and depth are adjustable through two parameters: the model layer factors (*depth_multiple*) and the model channel factors (*width_multiple*). Among these, YOLOv5s represents the variant with the minimal depth and width, boasting lower hardware requirements, ease of deployment, and superior training and detection speeds in comparison to its counterparts. However, due to its simplified network structure, YOLOv5s has a constrained feature extraction capability, leading to the lowest detection accuracy within the series, thereby highlighting significant potential for enhancement. As observed in Fig. 1, the structure of YOLOv5s is segmented into four modules: Input, Backbone, Neck and Head.

B. Input of YOLOv5s

The input module primarily processes the input image. During training, YOLOv5s employs Mosaic data augmentation techniques to enhance both the training efficiency and model accuracy. Additionally, it introduces methods for adaptive anchor box calculation and adaptive image scaling.

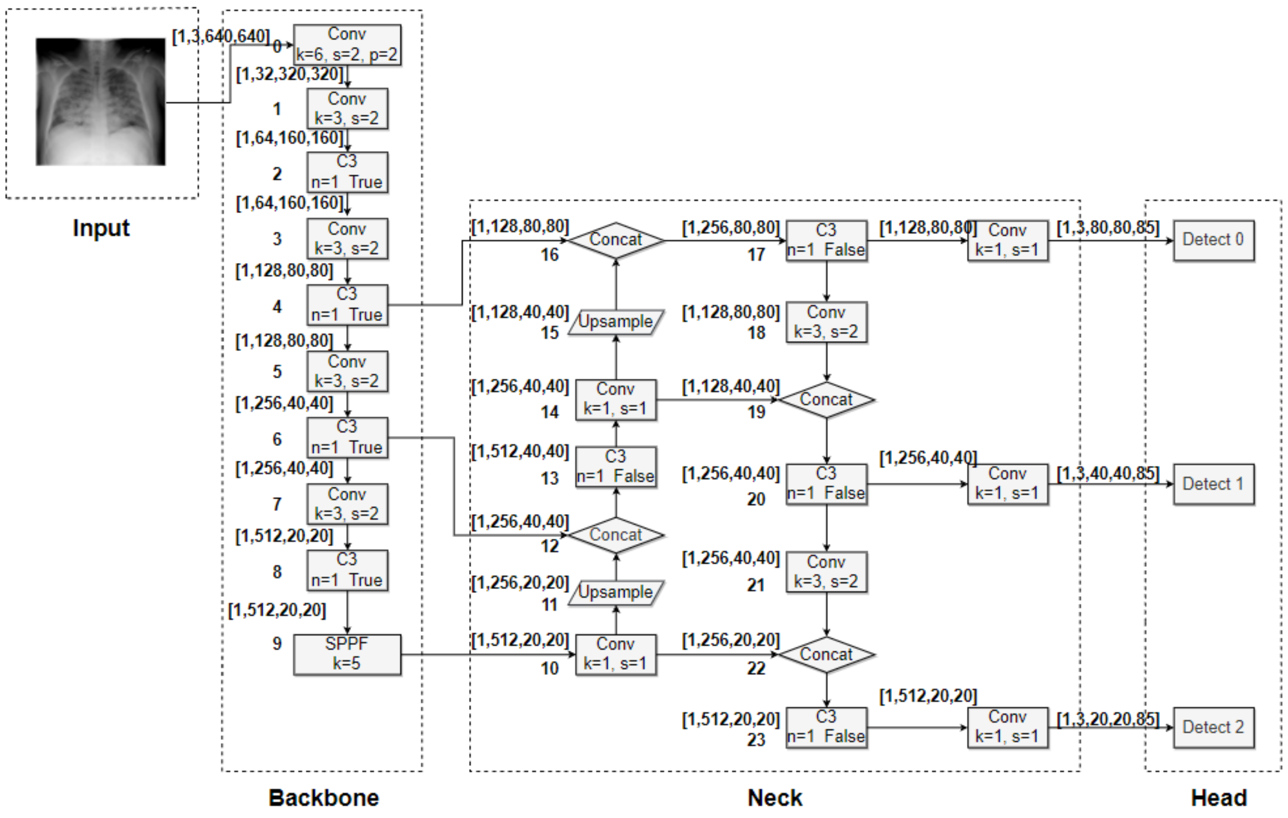


Fig. 1 The structure of YOLOv5s

Mosaic enhancement augments data from the initial two images to four, implementing random scaling, cropping, and arrangement of images. This data augmentation technique addresses the imbalance of small, medium, and large target data within the dataset. By randomly merging four images into one, it significantly enriches the dataset and enhances the network's robustness [12]. Concurrently, it also minimizes GPU usage: the random splicing method enables the processing of four images' data with a single image, thus reducing the batch size. This efficiency allows for improved outcomes even with a single GPU.

Adaptive anchor frame calculation generates anchor frames tailored to the input image's feature map size and ratio, optimizing the current feature map's fit. In the YOLO series algorithms, network training begins with a prediction analysis using a prior box, followed by the output of the predicted box, which is then compared to the label box. Subsequently, a backward update operation adjusts the network's parameters. Unlike YOLOv3 and YOLOv4, which calculate the initial anchor box for different datasets using a separate script, YOLOv5s integrates this functionality into the training code. Therefore, it adaptively computes anchors for varying datasets before each training session begins.

Common target detection algorithms must account for varying image dimensions, traditionally scaling images to a uniform standard size before processing. However, this approach can introduce different extents of black edges due to varying aspect ratios, leading to information redundancy and potentially slowing down the algorithm's inference speed. To address this, the YOLOv5s algorithm introduces a method that minimally adds black edges to the original image, thereby reducing computational load and enhancing the speed of target detection.

C. *Backbone of YOLOv5s*

The backbone network layer of YOLOv5s consists of the Focus module, Cross Stage Partial (CSP) module, and Spatial Pyramid Pooling Fast (SPPF) module.

The YOLOv5s algorithm integrates the Focus module at the network's initial input layer, facilitating the preliminary processing of input features. It achieves this by merging adjacent pixels, reducing the number of channels, and performing subsampling through a convolution operation with a stride of 2. This method efficiently preserves essential feature information while reducing computational demands, laying a strong foundation for further feature extraction.

Whereas the YOLOv4 architecture includes a single CSP structure within its backbone, YOLOv5s introduces two distinct CSP configurations: CSP1_X for the backbone and CSP2_X for the Neck network. The input is split into two pathways: one processes through CBS and multiple residual structures before reconvening, and the other convolves directly. These branches are then concatenated, passed through batch normalization, activated, and conclude with a CBS. As shown in Fig. 2, CSP1_X [13], designated for the backbone network, enhances the extraction of detailed features by bolstering the residual structure. This increment in residual layers prevents gradient loss due to network depth, ensuring robust feature extraction without the risk of network degradation. As shown in Fig. 3, CSP2_X [14], in contrast to CSP1_X, substitutes the Res unit with a 2*X CBS configuration, specifically catering to the Neck network. This adjustment tailors the structure for optimal feature integration within the Neck networks.

The YOLOv5s algorithm substitutes the Space Pyramid Pool (SPP) with the more efficient SPPF structure, incorporating it into the backbone network's final layer. This change augments the SPP's functionality. Illustrated in Fig. 4, the SPPF employs various sizes of pooled feature maps for subsampling. Following this, each subsampled feature map is concatenated, allowing for the generation of a uniform-dimensional feature map after SPP, while also amalgamating diverse pooling data. This approach not only maintains dimensional consistency post-processing but also enriches the feature map with varied pooling characteristics.

The YOLOv5s algorithm replaces the Space Pyramid Pool (SPP) with a more efficient SPPF structure, integrating it into the final layer of the backbone network. This modification enhances the effectiveness of SPP. The SPP

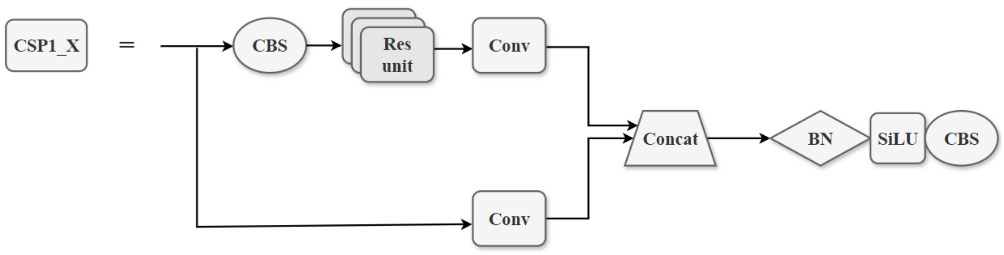


Fig. 2. The structure of CSP1_X

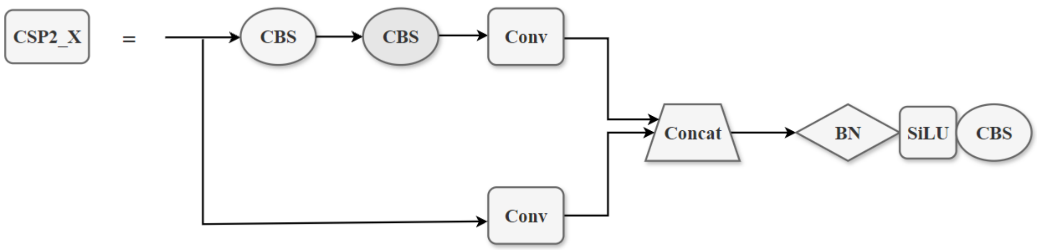


Fig. 3. The structure of CSP2_X

structure is illustrated in Fig. 5. By employing diverse sizes of pooled check feature maps for subsampling, and subsequently concatenating each subsampled feature map, it enables the acquisition of a feature map with identical dimensions post-SPP while simultaneously integrating distinct pooling information.

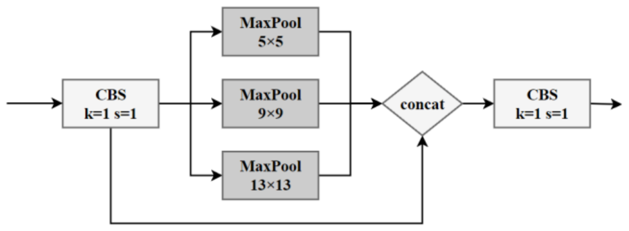


Fig. 4. The structure of SPP

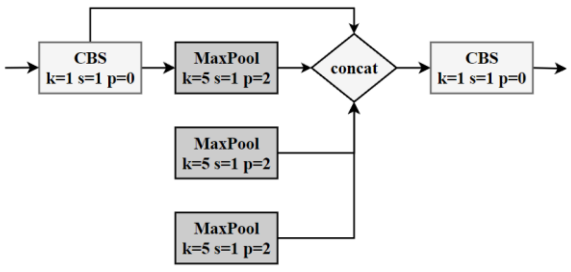


Fig. 5. The structure of SPPF

D. Neck of YOLOv5s

In the YOLOv5s object detection framework, the Neck module is instrumental in merging feature maps of varying levels to create a multi-scale feature map, enhancing target detection accuracy. The Neck structure in YOLOv5s employs the FPN+PAN combination [15], similar to YOLOv4, but introduces enhancements. Unlike YOLOv4's standard convolution operations, YOLOv5s adopts the CSP2 structure from CSPNet, improving network feature fusion capabilities.

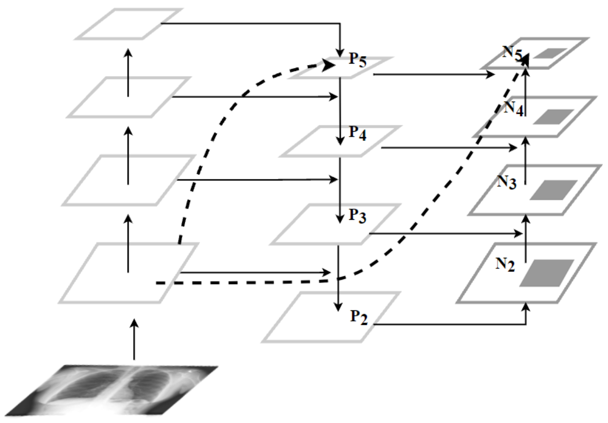


Fig. 6. The structure of FPN+PAN

The FPN+PAN architecture, depicted in Fig. 6, features a top-down approach in FPN by element-wise merging of higher-level semantic and lower-level information through upsampling. Conversely, the PAN architecture, building upon FPN, integrates lower-level localization details into deeper features, bolstering feature representation across various scales. The integration of top-down and bottom-up feature maps culminates in the final feature map for target detection.

E. Head of YOLOv5s

In the YOLOv5s algorithm, the head component is responsible for multi-scale target detection using the feature map derived from the backbone network. Pre-training involves acquiring coordinates for all target frames within the dataset, typically through K-means clustering to categorize training set target frames and identify several anchor frames. Generally, these anchor frames are grouped in threes, each corresponding to detection layers at different scales. The bounding box regression loss function predominantly utilizes the Intersection over Union (*IoU*) ratio, with the *IoU* calculation formula presented in equation (1).

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2} \tag{1}$$

The *IoU* compares the obtained prediction box and the real box, and uses the distance between the two boxes to reflect the detection effect of the model, as shown in Fig. 7. Where B_1 is the prediction box, B_2 is the true box, and B_3 is the smallest bounding rectangle containing B_1 and B_2. However, *IoU* only focuses on the position information of the prediction box and the real box. When the prediction box and the real box do not intersect, the value of *IoU* is 0, and there is no gradient in the loss function, thus learning and training cannot be carried out.

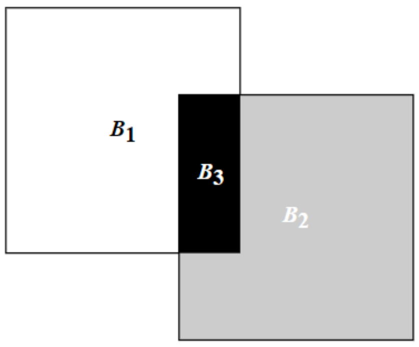


Fig. 7. The *IoU* diagram

The *CIoU Loss* is employed as the bounding box regression loss function at the output stage of YOLOv5s. This loss function considers the distance between the centers of the bounding boxes as well as their width and height, providing a more nuanced approach to addressing targets with significant aspect ratio disparities. It is particularly effective in refining the position and dimensions of bounding boxes for smaller targets. In comparison to traditional *IoU* Loss, *CIoU* Loss offers enhanced accuracy in target localization and ensures more consistent training performance in object detection tasks. The formula for *CIoU* Loss is depicted in equation (2).

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(B_1, B_2)}{c^2} + \beta v \tag{2}$$

Where $\rho(B_1, B_2)$ denotes the Euclidean distance between the center point of B_1 and B_2, c^2 represents the diagonal distance across B1 and B2's smallest enclosing rectangle, β is an tunable parameter, and v serves as a correction term.

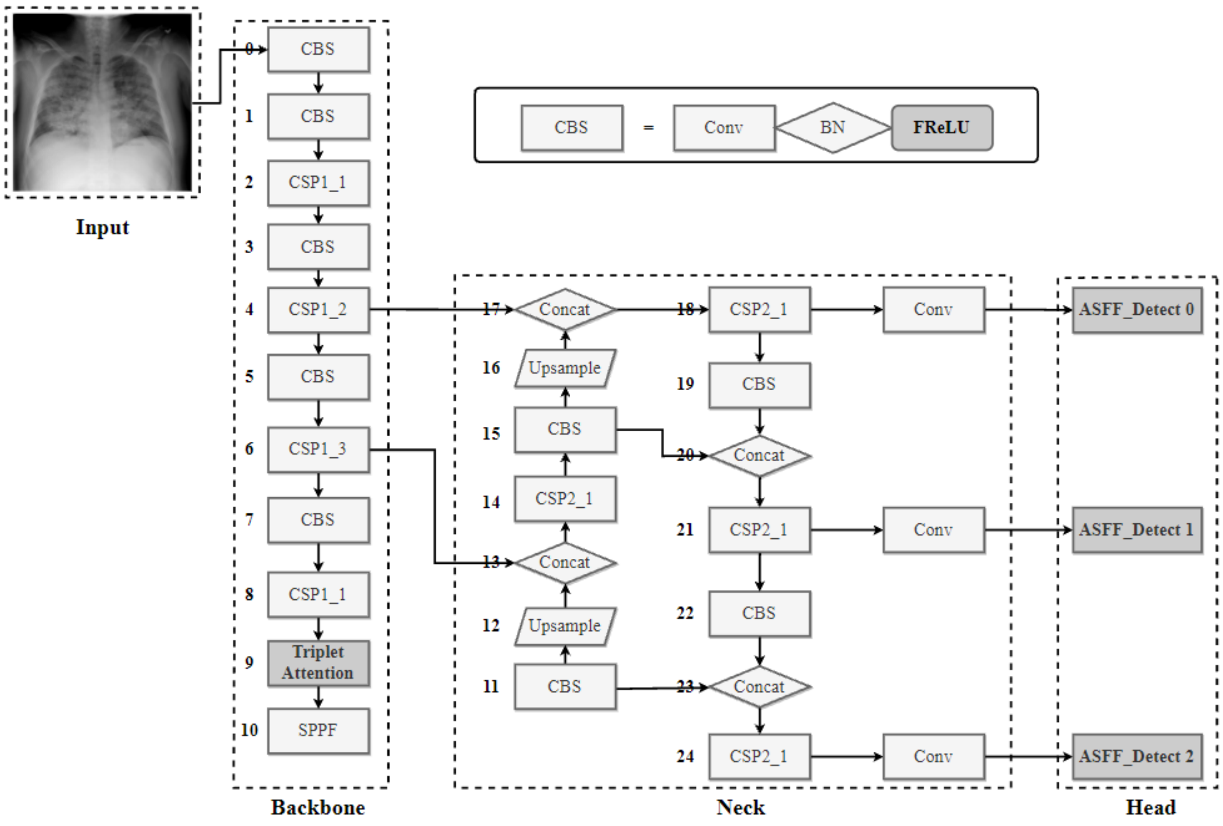


Fig. 8. The structure of YOLOv5-TAF

III. The Proposed Algorithm

This section delves into a variety of techniques and enhancement strategies. Initially, it outlines the detection methodology of the YOLOv5-TAF algorithm. It then introduces the Triplet attention mechanism, contrasts it with the CBAM attention mechanism, and elucidates the benefits of the Triplet attention mechanism over traditional channel attention mechanisms. Furthermore, the adaptive spatial feature fusion mechanism is discussed. Additionally, optimizations to the activation function are presented; *FReLU*, as compared to the original *SiLU* activation function in YOLOv5s, augments the activation space's sensitivity.

A. *YOLOv5-TAF Process Detection*

The YOLOv5-TAF algorithm processes images at a consistent size of 640×640×3. The integration of the Triplet attention mechanism into the Backbone significantly reduces the loss of channel and spatial interdependence that typically occurs when calculating channel attention at the single-pixel level, thereby enhancing detection accuracy without compromising operational speed. Table I details the backbone network parameters within the YOLOv5-TAF algorithm, highlighting the incorporation of the Triplet attention mechanism at the backbone's 9th layer. The ASFF is then introduced at the head stage to autonomously learn spatial weights for each scale, improving the amalgamation of features from various layers. By substituting the *SiLU* activation function with *FReLU* in the YOLOv5s algorithm, the system gains an improved ability to recognize complex patterns through enhanced pixel-level modeling.

The overarching architecture of the YOLOv5-TAF algorithm is depicted in Fig. 8, outlining the principal phases in Table I:

TABLE I
BACKBONE NETWORK PARAMETER

Layers	Module	Input size	Number	Args
1	Conv	640×640×3	1	[3,32,6,2,2]
2	Conv	320×320×32	1	[32,64,3,2]
3	C3	160×160×64	1	[64,64,1]
4	Conv	160×160×64	1	[64,128,3,2]
5	C3	80×80×128	2	[128,128,2]
6	Conv	80×80×128	1	[128,256,3,2]
7	C3	40×40×256	3	[256,256,3]
8	Conv	40×40×256	1	[256,512,3,2]
9	C3	20×20×512	1	[512,512,1]
10	Triplet	20×20×512	1	[512]
11	SPPF	20×20×512	1	[512,512,5]

Step 1: Initially, a chest X-ray image of dimensions 640×640×3 is inputted. This is followed by feature extraction using the CSPDarknet53 network, yielding four layers of feature maps with dimensions of 160×160×64, 80×80×128, 40×40×256, and 20×20×512 respectively. The final layer of feature maps is then processed through the Triplet attention mechanism before being fed into the SPPF

module.

Step 2: The SPPF module processes the final layer's feature map, which is then advanced to the feature pyramid fusion stage. At this stage, the first layer, consisting of a 20×20 feature map, is up-sampled and fused with the 40×40 feature map obtained in Step 1 through channel connection and feature fusion, producing a second-layer feature map. This process is repeated with the 80×80 feature map from Step 1 to generate the third-layer feature map. Following feature fusion, the three feature maps proceed to the multi-scale prediction branch, resulting in three distinct types of feature maps: 20×20×85, 40×40×85, and 80×80×85.

Step3: Anchor frames are assigned to the three scaled feature maps based on size. Subsequently, the model predicts the bounding box's center point coordinates, width and height offsets, the probability of containing an object, and the class score, utilizing the anchor box information. Through fitting predicted values to actual label values and employing a loss function for backpropagation and gradient descent training, the network parameters are iteratively updated to determine the final weight values.

B. *Triplet Attention*

The schematic of the proposed Triplet attention mechanism is depicted in Fig. 9. True to its name, Triplet attention features three parallel branches, with two dedicated to facilitating cross-dimension interaction between the channel dimension C and either the spatial dimension H or W. The third branch, akin to the CBAM approach, is designed for generating spatial attention. The outputs of these branches are combined through a simple averaging process.

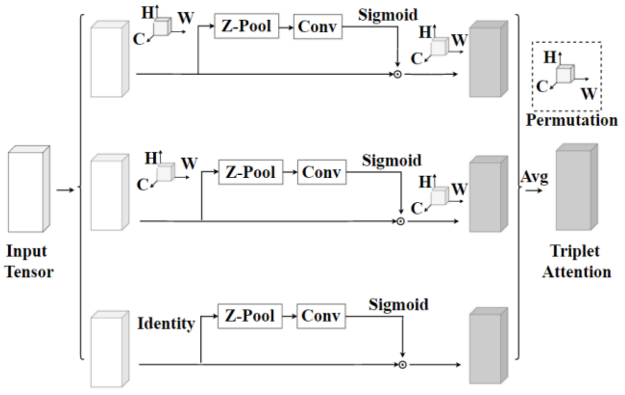


Fig. 9. The Triplet attention

Conventional approaches to channel attention computing typically contain scaling the feature maps in accordance with the singular weights calculated for each channel of the input tensor. Although efficient and effective, this method overlooks a critical aspect of spatial information preservation. Normally, channel weights are obtained by using global average pooling to reduce the input tensor to one pixel per channel, which results in an enormous loss of spatial information. In channel attention computing, this diminishes the relationship between channel and spatial dimensions. While the CBAM framework introduces a dual focus on Spatial and Channel attention to mitigate spatial interdependence issues, it treats channel and spatial attentions as distinct and independently computed entities. A comparative analysis of the Triplet attention mechanism and CBAM is presented in Fig. 10.

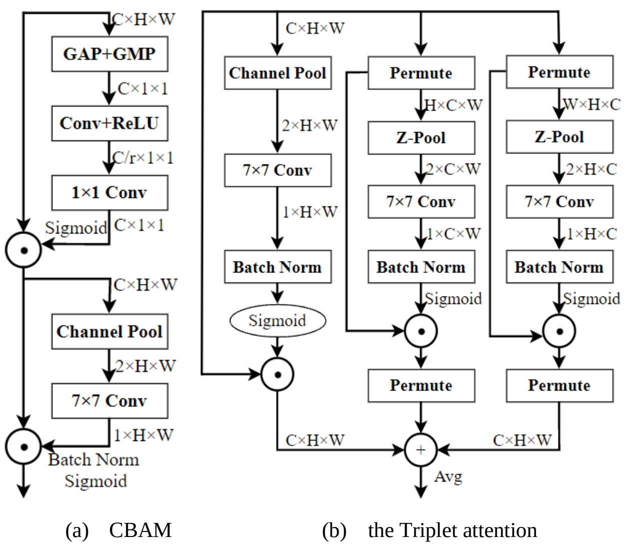


Fig. 10. Comparisons with different attention modules.

The Z-pool layer aims to condense the zeroth dimension of a tensor to two by amalgamating the features obtained from average and max pooling over the relevant dimension,. This process reduces its depth to streamline further computations. It can be represented by the (3):

$$Z-\text{pool}(x)=[MaxPool_{0d}(x), AvgPool_{0d}(x)] \tag{3}$$

where d signifies the dimension across which the pooling operations are executed.

Leveraging the operations outlined above, triplet attention is conceptualized as a three-branched module that processes an input tensor to produce a refined tensor retaining the original shape. For an input tensor x∈$R(C\times H\times W)$, the first branch enables interaction between the height H and channel C, facilitating an exchange of dimensions within the input feature map. This exchange involves linking the average and max pooling feature maps across a dimension, processing through a conventional convolutional and BN layer, followed by the computation of attention weights using the Sigmoid activation function for dimension exchange. The process concludes by reinstating the original dimension order of the input. The second branch initiates an interaction between the channel C and width W, with dimension exchange operations mirroring the first branch, including subsequent pooling. The third branch orchestrates an interaction between width W and height H, where the input feature's dimensions are permuted to ($W\times H\times C$), followed by pooling to reduce the tensor's channel dimensions to two. Ultimately, by applying an appropriate scaling factor, the outputs from all three branches, each maintaining the ($C\times H\times W$) shape, are amalgamated as depicted in equation (4) :

$$y=\frac{1}{3}\left(\overline{\hat{x}_1\sigma\left(\psi_1\left(\hat{x}_1^*\right)\right)}+\overline{\hat{x}_2\sigma\left(\psi_2\left(\hat{x}_2^*\right)\right)}+x\sigma\left(\psi_3\left(\hat{x}_3\right)\right)\right) \tag{4}$$

where σ denotes the *sigmoid* activation function; ψ_1, ψ_2, and ψ_3 represent the specialized two-dimensional convolutional layers with a kernel size k within the three branches of the triplet attention. Simplifying equation (4), y is derived as follows:

$$y=\frac{1}{3}\left(\overline{\hat{x}_1\omega_1}+\overline{\hat{x}_2\omega_2}+x\omega_3\right)=\frac{1}{3}\left(\overline{y_1}+\overline{y_2}+y_3\right) \tag{5}$$

Where ω_1, ω_2 and ω_3 are the cross-dimensional attention weights computed in the triplet attention mechanism. The $\overline{y}_1$ and $\overline{y}_2$ in (3) indicate a 90-degree clockwise rotation.

Compared to previous channel attention mechanisms, the triplet attention mechanism offers two advantages. First, it facilitates the capture of rich, discriminative feature representations with minimal computational overhead. Second, it underscores the significance of cross-dimensional interactions without resorting to dimensionality reduction, thus maintaining a direct correlation between channels and weights.

C. Adaptively Spatial Feature Fusion (ASFF)

The Feature pyramid (FPN) is a widely adopted strategy for addressing the challenge of scale variability in target detection. For single-shot detectors, a primary drawback is the inconsistency across different feature scales, which can hinder gradient calculation during training and diminish the feature pyramid's effectiveness. Direct feature fusion may also result in the loss of valuable information. This study introduces a novel, data-driven approach to pyramid feature fusion, termed adaptively spatial feature fusion (ASFF) [17]. This technique allows the network to autonomously learn to filter out irrelevant information from other feature levels, ensuring that only pertinent data is combined. Specifically, features from different levels are initially integrated and aligned to the same resolution. This is followed by training to identify the optimal fusion strategy. Features from various levels are adaptively merged at each spatial location. Unlike previous methods that summation or concatenation relied on element for integrating multi-level features, ASFF's core innovation lies in adaptively learning the spatial weight distribution for feature map fusion across scales, involving two key phases: uniform rescaling and adaptive fusion.

Because the features at four levels in YOLOv5s possess varying resolutions and channel counts, we customize each scale accordingly on sampling and sampling techniques. For up-sampling, a 1×1 convolution layer is first employed to reduce the feature channels to the corresponding level's count, followed by resolution enhancement through interpolation. In the case of down-sampling by a 1/2 ratio, a 3×3 convolution layer with a stride of 2 is utilized to concurrently adjust both the channel count and resolution. For a scale ratio of 1/4, a 2-stride max pooling layer precedes the 2-stride convolution.

Adaptive fusion refers to the fusion involves combining resized feature maps to create final fusion features. The fusion process for level l is described in (6), where $x^{n\to l}_{ij}$ represents the feature vector at the position (i, j) on the feature maps resized from level n to level l. y^l_{ij} represents the vector y^l of the output feature map. α^l_{ij}, β^l_{ij} and γ^l_{ij} indicate the spatial attention weights of feature maps at different levels.

$$y^l_{ij}=\alpha^l_{ij}\cdot x^{1\to l}_{ij}+\beta^l_{ij}\cdot x^{2\to l}_{ij}+\gamma^l_{ij}\cdot x^{3\to l}_{ij} \tag{6}$$

Let $\alpha^l_{ij}+\beta^l_{ij}+\gamma^l_{ij}=1$ and α^l_{ij}, β^l_{ij}, $\gamma^l_{ij}\in[0,1]$, and define α^l_{ij} in (7):

$$\alpha^l_{ij}=\frac{e^{\lambda^l_{\alpha ij}}}{e^{\lambda^l_{\alpha ij}}+e^{\lambda^l_{\beta ij}}+e^{\lambda^l_{\gamma ij}}} \tag{7}$$

Where $\lambda^l_{\alpha ij}$, $\lambda^l_{\beta ij}$ and $\lambda^l_{\gamma ij}$ are the control parameters.

ASFF offers the flexibility to accommodate feature maps of varying sizes and resolutions, making it highly adaptable to different data types and tasks. Through learning, ASFF dynamically assigns weights to various feature maps, enabling the network to tailor feature fusion to specific tasks and thereby enhance its representational capabilities. It is straightforward to implement and incurs minimal computational costs. Overall, as a feature fusion module, ASFF plays a crucial role in deep neural networks by aiding in the capture of spatial information and enhancing the modeling of data such as images.

D. Funnel activation (FReLU)

FReLU, or Funnel Rectified Linear Unit [18], builds upon the concept of *ReLU*/*PReLU* by incorporating a spatial condition, as shown in Fig. 11. This addition is both simple to implement and contributes only a marginal increase in computational demand. Formally, the method is defined as $y=max(x,T(x))$, where $T(x)$ is a straightforward and effective spatial contextual feature extractor. By integrating a spatial condition into activation functions, *FReLU* effectively extends *ReLU* and *PReLU* [19] to a visually parametric *ReLU* with pixel-wise modeling capabilities.

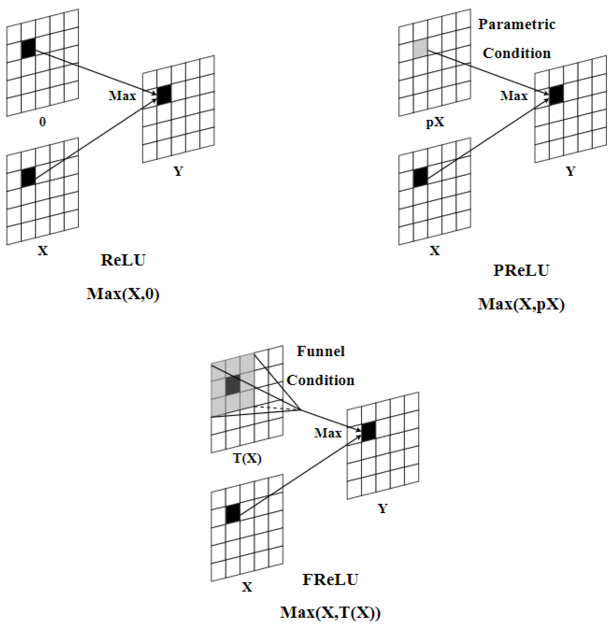


Fig. 11. The funnel activation

FReLU utilizes the same $max(\cdot)$ function for its non-linear operation. However, for the conditional part, *FReLU* enhances this by incorporating a 2D spatial context-dependent condition for each pixel, diverging from more recent methods that rely on conditions based on the pixel itself or its channel context. It employs a $max(\cdot)$ operation to determine the maximum between x and a given condition. Formally, the funnel condition is defined as $T(x)$.

To apply the spatial condition, a Parametric Pooling Window is used to establish spatial dependency. Specifically, the activation function is defined as follows:

$$f\left(x_{c,i,j}\right)=\max\left(x_{c,i,j},T\left(x_{c,i,j}\right)\right) \tag{8}$$

$$T\left(x_{c,i,j}\right)=x_{c,i,j}^{\omega}\cdot p_{c}^{\omega} \tag{9}$$

Where $x_{c,i,j}$ is the input pixel for the non-linear activation $f(\cdot)$ on the c-th channel, at the 2-D spatial position (i, j); function $T(\cdot)$ represents the funnel condition, $x^{\omega}_{c,i,j}$ symbolizes a $k\times k$ Parametric Pooling Window centered on $x_{c,i,j}$, p^{ω}_{c} denotes the coefficient on this window which is shared in the same channel, and $(\cdot)$ indicates dot multiplication.

This definition of the funnel condition enables the network to introduce spatial conditions into the non-linear activations for each pixel, conducting non-linear transformations and creating spatial dependencies simultaneously. Unlike typical approaches that generate spatial dependencies in the convolution layer and perform non-linear transformations separately, the funnel condition integrates these aspects. Normally, activations are not explicitly dependent on spatial conditions; the funnel condition changes this, making them so.

As a result, the pixel by pixel condition endows the network with pixel-wise modeling capacity, where the $max(\cdot)$ enables each pixel to choose whether to consider the spatial context. Formally, consider a network $\{F_1, F_2, ..., F_n\}$ with n *FReLU* layers, each layer has a parametric window. After the F_n layer, this set expands to $\{1,1+r,1+2r,...,1+nr\}$, offering increased choices for each pixel and enabling the approximation of any layout, given a sufficiently large n. Given that the layout of objects in images often favors oblique lines or arcs instead, the spatial structure of objects is naturally captured through the pixel-wise modeling capacity provided by the spatial condition.

IV. EXPERIMENT

A. Experimental Data

The utilized in this study comprises chest X-rays of children aged 1-5, collected by Kermany et al. at the University of California, San Diego, and the Guangzhou Women and Children Medical Center in China. The dataset includes 5856 images, with 4273 X-ray images of pneumonia and 1583 images of normal lungs. The distribution of these two types of chest X-rays within the dataset is detailed in Table II.

TABLE II
ALLOCATION TABLE FOR DIFFERENT TYPES OF IMAGES IN DATA SET

Chest X-Ray Images	train	val	total
pneumonia	3883	390	4273
normal	1349	234	1583
total	5232	624	5856

The Fig. 12 displays images from the dataset, where (a) represents a normal lung and (b) depicts a lung with pneumonia.

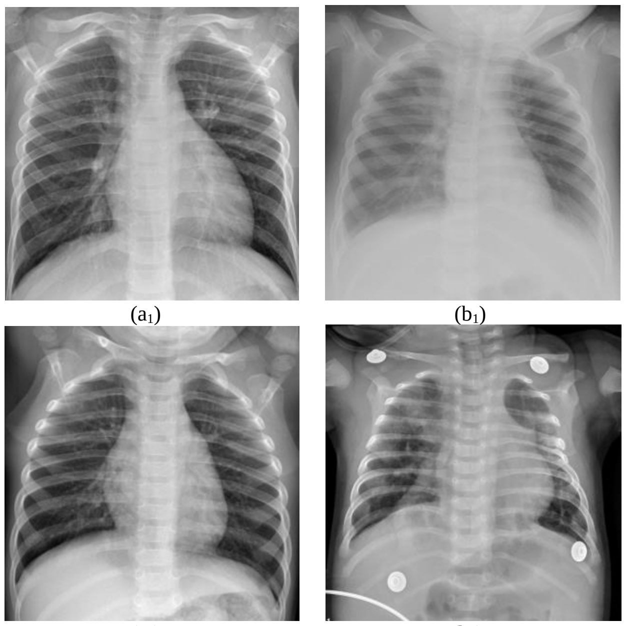


Fig. 12. The X-Ray image of lungs. (a)normal; (b)pneumonia

B. Experimental Settings

In this experiment, the dimensions of all input lung images were standardized to 640×640×3. The training was conducted over 400 epochs, with a batch size of 16. The initial learning rate was established at 0.001, and the momentum term was fixed at 0.9. These configurations are detailed in Table III.

TABLE III
EXPERIMENTAL ENVIRONMENT CONFIGURATION TABLE

Configuration	Configuration parameter
Operating System	Windows 10
CPU	14 vCPU Intel (R) Xeon (R) Gold 6330 CPU @ 2.00 GHz
GPU	RTX 3090
Memory	24GB
Deep learning framework	Pytorch1.10.2
CUDA version	Cuda11.4
Integrated development environment	PyCharm

C. Performance Indicators

In this study, precision, recall, *mAP*@0.5, frames per second (*FPS*), number of parameters and model size were used as evaluation indicators.

Precision quantifies the fraction of accurately identified positive cases in the pneumonia detection task relative to all cases labeled as positive. The formula is presented below:

$$\text{Precision}=\frac{TP}{TP+FP} \tag{10}$$

Recall measures the ratio of correctly predicted positive cases to the total number of actual positive cases in the test set. The formula is described as follows:

$$\text{Recall}=\frac{TP}{TP+FN} \tag{11}$$

where *TP* represents the number of true positives, *FP* represents the number of false positives, *FN* represents the number of false negatives, and *TN* represents the number of true negatives.

The *AP* calculates the average precision across all classes in a single-class model, with *P* denoting precision as the vertical axis and *R* denoting recall as the horizontal axis on the Precision-Recall (*P-R*) curve. The area under the curve relative to the axis represents the *AP* value. The formula is detailed below:

$$AP = \int_0^1 P(R)\,dR \tag{12}$$

The *mAP* is the mean average precision aka. The formula is as follows:

$$mAP = \frac{\sum_{i=0}^{n} AP_i}{n} \tag{13}$$

Where *n* represents the total number of categories and AP_i represents the *AP* value of class *i*.

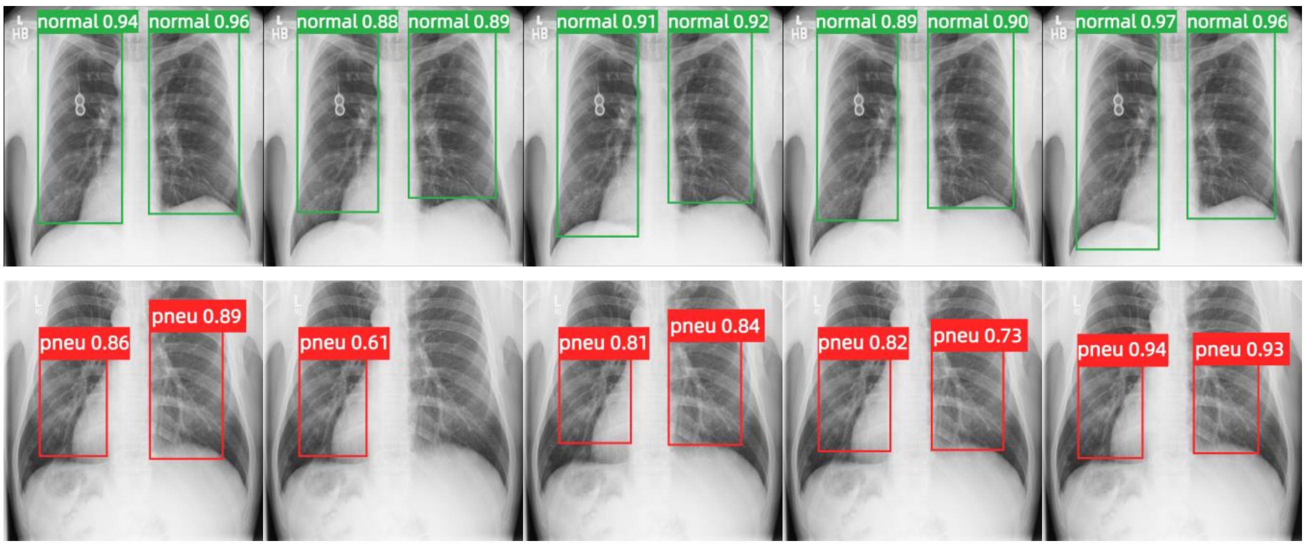
D. Analysis of experimental results

To further validate the advantages of the proposed YOLOv5-TAF algorithm, this study compares it with Faster-RCNN, SSD, and YOLOv8s algorithms. The comparison results are presented in Table IV. As indicated in Table IV, compared to two-stage algorithms like Faster RCNN, one-stage algorithms, including the SSD and YOLO series, offer faster detection speeds. Notably, YOLOv5-TAF demonstrates improved detection accuracy over both SSD and YOLOv8s.

The Fig. 13 illustrates the detection performance of these algorithms on two sets of images. It is observed that the SSD algorithm experiences significant positioning errors, leading to false positives and missed detections, which are attributed to its lack of feature fusion across different levels. While the Faster RCNN algorithm shows fewer false positives and missed detections, its two-stage nature results in slower detection speeds. Although YOLOv8s represents the latest in the YOLO series from Ultralytics, its performance does not surpass that of YOLOV5-TAF for the

TABLE IV
COMPARISON OF THE DETECTION EFFECT OF EACH MODEL

Object Detection Algorithm	Model Size /MB	Params (M)	FPS	Precision	Recall	mAP
Faster RCNN	108.2	13589372	21.45	71.73	84.06	87.38
SSD	100.3	813184	56.23	82.47	81.66	86.33
YOLOv5s	54.2	7129567	59.83	83.17	84.29	88.14
YOLOv8s	66.11	1866262	65.23	82.09	81.24	82.82
YOLOv5-TAF	55.4	36860543	58.72	88.82	86.55	91.71



(a) Faster-RCNN algorithm (b) SSD algorithm (c) YOLOv5s algorithm (d) YOLOv8s algorithm (e) YOLOv5-TAF algorithm

Fig. 13. The detection effect of different algorithms.

dataset used in this experiment. Hence, this experiment builds upon the foundation of YOLOV5-TAF.

Within the YOLOV5-TAF framework, varying input sizes impact *mAP* and detection speed. Table V reveals that an input size of 1280×1280 yields the highest detection quality but significantly slows down the process, hindering experimental progress. Conversely, a 320×320 input size accelerates speed but at the cost of a lower *mAP*. An input size of 640×640 strikes an optimal balance between detection quality and speed, making it the preferred choice.

TABLE V
INFLUENCE OF DIFFERENT INPUT SIZES ON THE DETECTION EFFECT

Input size	Speed (FPS)	Epoch	mAP
320×320	78.67	100	87.73
640×640	58.48	100	88.14
1280×1280	30.42	100	88.39

E. Ablation Experiment

To validate the effectiveness of each module within the YOLOV5-TAF algorithm, ablation studies were conducted using the same dataset and identical training hyperparameters. Additionally, incremental enhancements were applied atop the YOLOv5s baseline to further augment the model's performance. The outcomes of these experiments are detailed in Table VI.

TABLE VI
ABLATION EXPERIMENT

Method	Triplet	ASFF	FReLU	mAP
YOLOv5s				88.14
Proposed method1	√			89.86
Proposed method2		√		90.75
Proposed method3			√	88.84
Proposed method4	√	√		91.53
Proposed method5	√		√	90.06
Proposed method6		√	√	91.0
Proposed method7	√	√	√	91.71

The Table VI highlights that YOLOv5s performed well various enhancement points. The integration of the Triplet Attention Module into the original YOLOv5s model enabled the capture of rich, discriminative features with minimal increase in computational complexity, resulting in a 1.72% increase in *mAP*@0.5. The addition of ASFF to enhance the feature pyramid further improved accuracy, with a 2.61% rise in *mAP*@0.5. Switching the activation function to *FReLU* led to a 0.70% increase in *mAP*@0.5. By amalgamating the strengths of each module, the algorithm ultimately achieved an *mAP*@0.5 of 91.71%.

The attention mechanism within the YOLOv5s algorithm was enhanced by incorporating the Triplet Attention mechanism into the backbone network, while other components remained unchanged. This variant was named YOLOv5-Tri. The loss curves for both YOLOv5s and YOLOv5-Tri are depicted in Fig. 14.

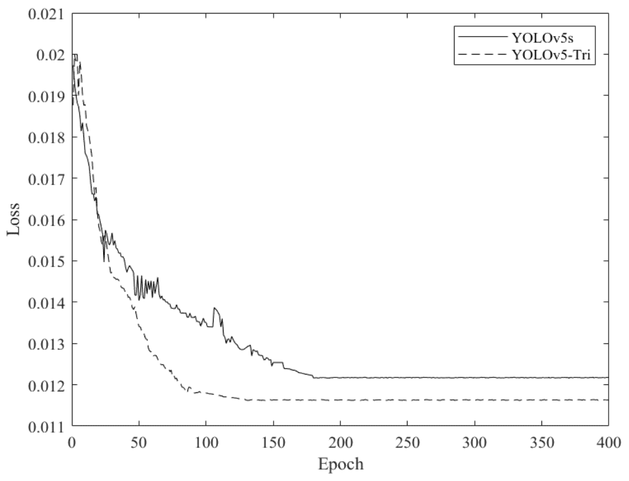


Fig. 14. The comparison of loss curves of YOLOv5s and YOLOv5-Tri.

As observed in Fig. 14, YOLOv5-Tri exhibited rapid convergence within the first 90 iterations, with a noticeable decline in convergence speed post approximately 100 iterations. The rate of convergence gradually decelerated between 100-120 iterations, with minimal changes in algorithm loss value after 150 iterations, culminating in a final loss of 0.0116. When compared to YOLOv5s, YOLOv5-Tri showed a marked improvement in convergence speed and a significant reduction in training time.

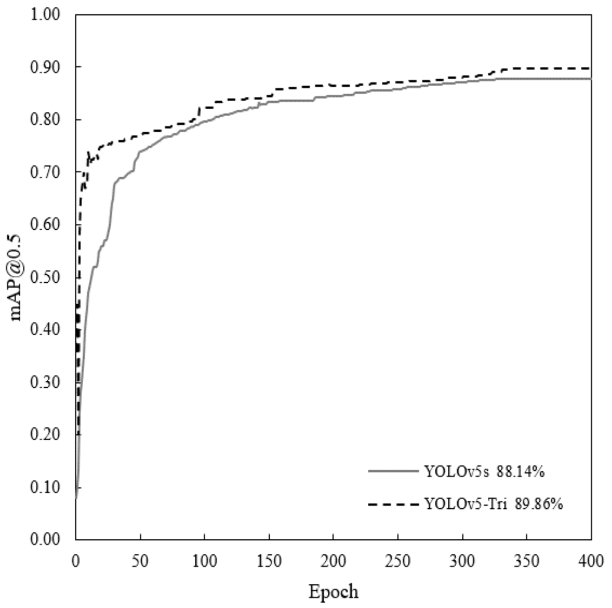


Fig. 15. The comparison of mAP@0.5 curves of YOLOv5s and YOLOv5-Tri

The Fig. 15 presents a comparison of the *mAP*@0.5 curves between the YOLOv5-Tri algorithm and the original YOLOv5s algorithm. As illustrated in Fig. 15, the *mAP*@0.5 value for the YOLOv5-Tri algorithm is 1.7% higher than that of the original YOLOv5s algorithm.

To further enhance detection accuracy, the ASFF module was integrated into the YOLOv5-Tri algorithm, resulting in the improved YOLOv5-TA algorithm. This modified algorithm was trained on the dataset, and its convergence curve is displayed in Fig. 16. The YOLOv5-TA algorithm demonstrates effective convergence, with the initial loss value reducing from 0.0116 to 0.0096 and a noticeable improvement in convergence speed. These results indicate that the ASFF module positively optimizes the network.

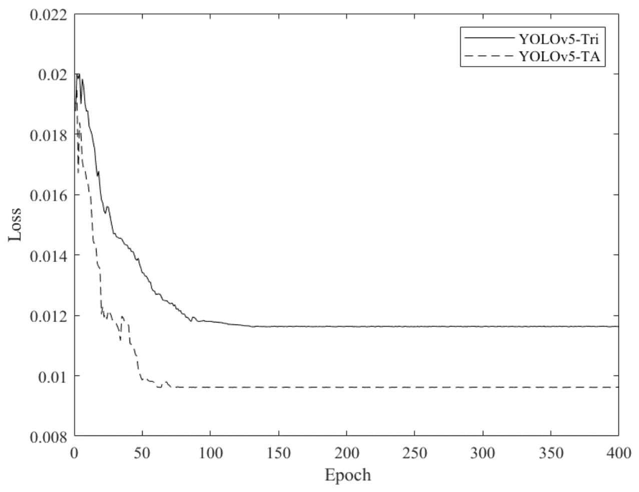


Fig. 16. The comparison of loss curves of YOLOv5-Tri and YOLOv5-TA

The Fig. 17 illustrates the mAP@0.5 curve comparison between the YOLOv5-TA algorithm and the YOLOv5-Tri algorithm on the dataset. With the inclusion of the ASFF module, the mAP@0.5 for the YOLOv5-TA algorithm reaches 91.53%, marking a 1.67% improvement over the YOLOv5-Tri algorithm and enhancing target detection capabilities.

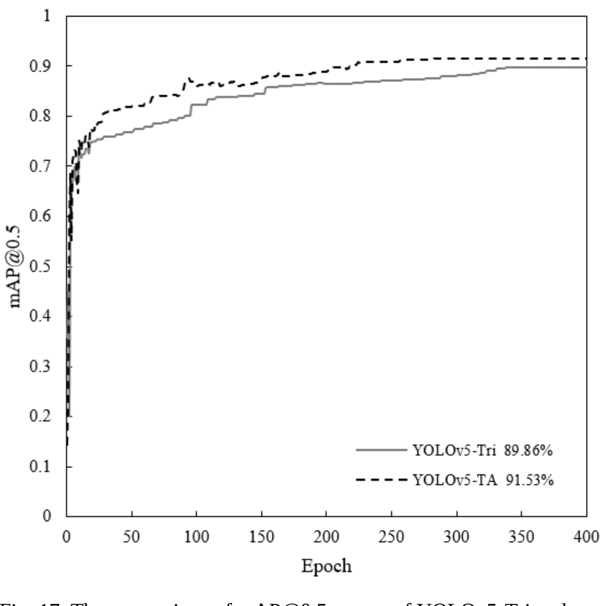


Fig. 17. The comparison of mAP@0.5 curves of YOLOv5-Tri and YOLOv5-TA

The Fig. 18 presents the *mAP*@0.5 curve comparison between the YOLOv5-TAF algorithm and the YOLOv5-TA algorithm. As shown in Fig. 18, the introduction of the *FReLU* activation function elevates the YOLOv5-TAF algorithm's *mAP*@0.5 to 91.71%, a 0.18% increase compared to the YOLOv5-TA algorithm.

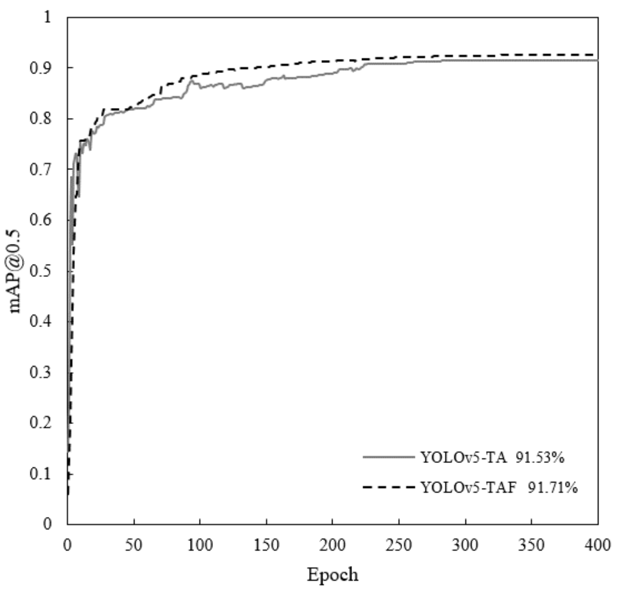


Fig. 18. The comparison of mAP@0.5 curves of YOLOv5-TA and YOLOv5-TAF

V. CONCLUSIONS

The YOLOv5-TAF algorithm introduced in this study addresses the challenge of detecting less conspicuous pneumonia lesions in X-ray images by incorporating a triplet attention mechanism within the backbone network. To more effectively highlight target features and enhance the network's capability to amalgamate features of targets across various scales, a feature fusion mechanism inspired by ASFF is proposed. Furthermore, substituting the *SiLU* activation function in the YOLOv5s model with *FReLU* facilitates the capture of complex patterns and boosts the sensitivity of the activation space through pixel-wise modeling. Experimental findings demonstrate that the YOLOv5-TAF algorithm enhances pneumonia detection. This research primarily focuses on accuracy enhancements while less emphasis is placed on detection speed; future efforts could explore the development of a lightweight deep learning model to expedite model detection speeds.

REFERENCES

[1] P. K. Wong, T. Yan, H. Wang, I. N. Chan, J. Wang, Y. Li and C. H. Wong. "Automatic detection of multiple types of pneumonia: Open data set and a multi-scale attention network," *Biomedical Signal Processing and Control*, vol. 73, pp. 103415, 2022.

[2] S. K. Zhou, H. Greenspan, C. Davatzikos, J. S. Duncan, A. Madabhushi and R. M. Summers. "A Review of Deep Learning in Medical Imaging: Imaging Traits, Technology Trends, Case Studies with Progress Highlights, and FaDSuture Promises," *Lecture Notes in Proceedings of the IEEE*, vol. 109, no. 5, pp. 820-838, 2021.

[3] R. Girshick, J. Donahue, T. Darrell and J. Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 23-28 June, 2014, Columbus, pp. 580-587.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2022.

[5] N. Dalai and B. Triggs, "Histograms of oriented gradients for human detection," *In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 20 - 26 June, 2005, San Diego, pp. 886-893.

[6] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154-171, 2013.

[7] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Lecture Notes in IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149, 2017.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proceedings of the IEEE Computer on Computer Vision and Pattern Recognition*, 26 June - 1 July, 2016, Las Vegas, pp. 779-788.

[9] D. Misra, T. Nalamada, A. U. Arasanipalai, and Q. Hou, "Rotate to attend: Convolutional triplet attention module," *In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 5-9 January, 2021, Waikoloa, pp. 3139-3148.

[10] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," *In Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10-17 October, 2021, Montreal, pp. 2778-2788.

[11] R. Qu, Y. Yang and Y. Wang, "COVID-19 detection using CT image based on YOLOv5 network," *Lecture Notes in 2021 3rd International Academic Exchange Conference on Science and Technology Innovation (IAECST)*, 10-12 December, 2021, Guangzhou, pp. 622-625.

[12] H. He, Z. Zhang, Q. Jia, L. Huang, Y. Cheng, and B. Chen, "Wildfire detection for transmission line based on improved lightweight YOLO," *Energy Reports*, vol. 9, no. 1, pp. 512-520, 2023.

[13] M. Huang, B. Wang, J. Wan and C. Zhou, "Improved blood cell detection method based on YOLOv5 algorithm," *2023 IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 24-26 February, 2023, Chongqing, pp. 992-996.

[14] X. He, R. Cheng, Z. Zheng, and Z. Wang, "Small object detection in traffic scenes based on YOLO-MXANet," *Sensors*, vol. 21, no. 21, pp. 7422, 2021.

[15] Z. G. Li, J. H. Zhang, B. Li, X. Y. Gu and X. D. Luo, "COVID-19 Diagnosis on CT Scan Images Using a Generative Adversarial Network and Concatenated Feature Pyramid Network with an Attention Mechanism," *Medical Physics*, vol. 48, no. 8, pp. 4334-4349, 2021.

[16] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," *Proceedings of the European Conference on Computer Vision (ECCV)*, 8-14 September, 2018, Munich, pp. 3-19.

[17] C. Zhang, J. Liu, J. Xiao and J. Xiong, "Water surface target detection and recognition of USV based on YOLOv5," *2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 27-29 May, 2022, Beijing, pp. 1227-1231.

[18] N. Ma, X. Zhang and J. Sun, "Funnel activation for visual recognition," *Lecture Notes in Computer Vision–ECCV 2020: 16th European Conference*, 23-28 August, 2020, Glasgow, pp. 351-368.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imageNet classification," *In Proceedings of the IEEE International Conference on Computer Vision*, 11-18 December, 2015, Santiago, pp. 1026-1034.