



Article

Balanced-YOLOv3: Addressing the Imbalance Problem of Object Detection in PCB Assembly Scene

Jing Li ^{1,2} , Yingqian Chen ³, Weiye Li ³ and Jinan Gu ^{1,*} ¹ School of Mechanical Engineering, Jiangsu University, Zhenjiang 212000, China; aygxyj@163.com² School of Electronic Information and Electrical Engineering, Anyang Institute of Technology, Anyang 455000, China³ School of Mechanical and Electrical Engineering, Guangdong University of Technology, Guangzhou 510006, China; qianrunfar@163.com (Y.C.); friendlymt@163.com (W.L.)

* Correspondence: gujinan@tsinghua.org.cn

Abstract: The object detection algorithm of the PCB (Printed Circuit Board) assembly scene based on CNN (Convolutional Neural Network) can significantly improve the production capacity of intelligent manufacturing of electronic products. However, the object class imbalance in the PCB assembly scene, the multi-scale feature imbalance, and the positive/negative sample imbalance in the CNN have become critical problems restricting object detection performance. Based on YOLOv3, this paper proposes a class-balanced Train/Val (Training set/Validation set) split method for object class imbalance, an additional feature fusion strategy for multi-scale feature imbalance, and an efficient anchor concept for positive/negative sample imbalance. These three contributions are Balanced-YOLOv3. After experimental verification, compared with other YOLOv3 series algorithms, the mAP@.5 (Mean Average Precision at Intersection over Union threshold 0.5) and mAP@.5:.95 (average mAP over different Intersection over Union thresholds, from 0.5 to 0.95, step 0.05) of Balanced-YOLOv3 have achieved the best results and ranked third in the metrics of parameter and inference time. Compared with other current anchor-based object detection algorithms, Balanced-YOLOv3 has excellent detection performance and low computational complexity, which effectively solves the problem of imbalanced object detection in PCB assembly scenarios.

Keywords: Balanced-YOLOv3; imbalance problem; object detection; class balanced split; efficient anchor; PCB assembly scene



Citation: Li, J.; Chen, Y.; Li, W.; Gu, J. Balanced-YOLOv3: Addressing the Imbalance Problem of Object Detection in PCB Assembly Scene. *Electronics* **2022**, *11*, 1183. <https://doi.org/10.3390/electronics11081183>

Academic Editor: Maciej Ławryńczuk

Received: 9 March 2022

Accepted: 3 April 2022

Published: 8 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Artificial intelligence (AI) is expanding from consumer intelligence to manufacturing intelligence [1–5]. In intelligent production, automated production plants are currently used more frequently. Benefiting from the advancement of machine vision technology, the application of AI in product quality monitoring and defect management is becoming more and more critical. Especially in the electronic product manufacturing industry, closely related to general public life, it is more difficult to produce and assemble due to the wide variety of electronic product parts and sizes, coupled with short product life cycles, high quality, and cost sensitivity. The vision-based PCB assembly scene object detection algorithm can identify and locate the PCB, holes to be assembled, assembled holes, and various electronic components in the picture [6]. These inspection results can be applied to many vital areas, such as product defects, AR (Augmented Reality) guiding assembly [7], generating assembly scene descriptions, and assisting robots in replacing humans in achieving intelligent assembly, etc., and can significantly improve the production capacity of electronic product companies in intelligent manufacturing.

As an advanced model structure, CNN can extract various feature information from images and at the same time use different layers of the network to express semantic and location information with different emphases, which can well solve the question of the

“what” and “where” of the object in the detection task. The rapid development of CNN-based visual object detection tasks came about because when CNN processes images, it can extract image features by itself through multiple convolution kernels and deal with object displacement scaling. Concerning other forms of distortion invariance, it has good robustness and computational efficiency. However, there are various imbalance problems in object detection, and these imbalances will affect the final detection effect. Many existing studies can be attributed to the methods to solve these imbalances [8]. Class imbalance refers to a situation where one or more classes are over-represented, which will cause the classifier to be biased against over-exposed classes during training. Feature scale imbalance refers to the significant difference in the performance of object features across different scale spaces, which will lead to poor representation of the object due to information fused across other scale features. The imbalance of positive and negative samples means that in the anchor-based object detection algorithm, an object can only be responsible for one anchor, and other anchors are considered negative samples. Too many negative samples will increase the inference time and affect the judgment of the classifier.

As a special kind of object detection, PCB assembly scene object detection also has imbalance problems that affect its detection accuracy and speed. In the object detection dataset of the PCB assembly scene, there are many electronic components to be assembled or assembled, and a small number of PCBs to be assembled and to be assembled through holes. The object size of the PCB assembly scene varies widely, from small through-holes, SMD (Surface Mount Device), and plug-in electronic components to large-sized PCBs. When these objects are extracted through the backbone, there will be multi-scale feature imbalance problems, and even a small-sized objects' location and semantic features disappear. At the same time, the YOLO series algorithm uses nine anchors evenly allocated to three detection heads without considering the spatial position and size of the actual PCB assembly scene object distribution. It cannot generate efficient anchors that adapt to the PCB assembly scene objects. There is a positive/negative sample imbalance based on dense anchors.

Aiming at the problem of class imbalance in object detection, the current research includes three main aspects: oversampling with few samples [9], under-sampling with multiple samples [10,11], and mixed sampling [12,13]. However, whether it is oversampling or under-sampling, it is necessary to add additional small-sample data or remove multi-sample data, which will inevitably add some redundant information or missing samples to the original data. To the best of our knowledge, no one has discussed the issue of object class balance from the perspective of the train-validation-test split. Aiming at the imbalanced scale of object features extracted by the object detection backbone network, the current research mainly includes the cascade method of shallow and deep features of the backbone network [14–17] and the unified size transformation method of feature fusion [18–20]. However, few people consider the fusion method of multi-scale imbalanced features according to the spatial distribution of the object to be detected. Regarding the imbalance of positive/negative samples in object detection, the current research includes three main aspects: balancing the number of positive/negative samples [21–25], attention mechanism [26–28], and weight focus loss function [29–34]. However, for anchor-based object detection algorithms, few people solve the problem of the anchor generation method, which is the source of positive /negative sample imbalance.

Although CNN-based object detection algorithms emerge in an endless stream, they have become a research and application hotspot in academia and industry because the one-stage anchor-based YOLO series algorithms are efficient and accurate. Meanwhile, due to limited computing resources and insufficient software support in various practical applications, YOLOv3 is still one of the most widely used detectors in the industry. Based on the above problem analysis and knowledge gaps, we choose YOLOv3 as the object detection framework, take PCB, THT (Through Hole Technology) electronic components, and through holes under two scales as the detection objects, solve the imbalance problem that affects the object detection effect as the breakthrough point. An object detection method

is designed in the PCB assembly scene with high recognition accuracy and high positioning accuracy. The main contributions of this paper are summarized as follows:

- (1) We propose a class-balanced Train/Val split method. This method ensures that all classes are included in Train/Val and that the proportion of the number of classes allocated in Train/Val is the same. The class ratio imbalance bias is caused by the random number of pictures split to Train/Val.
- (2) We construct an addition feature fusion model suitable for object detection problems in PCB assembly scenarios. This feature fusion method can preserve the invariant and equivariant information of multi-scale objects, speeding up the training and learning process by reducing the number of channels and making up for the disappearance of information brought by small-size objects through deep networks.
- (3) We design an efficient anchor, which analyzes the size of the sample labels in the training set by clustering, and counts the spatial distribution on the three output ports of the detection head, which determines the number of anchors with actual detection effect in the data set. The efficient anchor alleviates the imbalance of positive/negative samples and directly provides the basis for the assignment of anchors.

The following section reviews object detection methods for PCB assembly scenarios and various approaches to address object detection imbalance. In Section 3, we describe the specific implementation details of the proposed method. In Section 4, we show the experimental results and the effectiveness of the proposed method. Finally, Section 5 concludes the paper.

2. Related Work

Since the proposed Balanced-YOLOv3 solves the imbalance problem in object detection in PCB assembly scenes, it involves class imbalance, multi-scale feature imbalance, and positive/negative sample imbalance. We summarize the research results related to the above imbalance problems and object detection in PCB assembly scenarios.

2.1. Class Imbalance in Object Detection

The related research work on class imbalance is summarized as follows. Ren et al. proposed BALMS (Balanced Meta-Softmax) for long-tailed visual recognition. They derived a balanced softmax function from a probabilistic point of view, which explicitly considered changes in label distribution during optimization. They introduced Meta-sampler, which resamples via meta-learning for high validation accuracy [9]. Wang et al. proposed that after an instance segmentation model is trained, a novel two-layer sampling scheme combining image-level and instance-level sampling is used to collect class-balanced proposals, and then these collected proposals are used to calibrate the classify head to improve the performance of the tail class. The scheme also contains a simple two-head inference component, which can effectively alleviate the performance degradation of the head class after calibration [10]. Wang et al. proposed the Dynamic Curriculum Learning (DCL) framework for imbalanced data learning and designed two-level curriculum schedulers. The first stage is a sampling scheduler that dynamically trains the model from imbalanced to balanced, and the second stage is a loss scheduler that combines cross-entropy loss (CE) and metric learning loss, thereby achieving a good balance between class bias accuracy and class balance accuracy [11]. In learning classification tasks, Kang et al. separated representation learning and classifier learning, readjusted the classification boundary, and proposed deconstructing the traditional “classifier representation joint learning” paradigm, seeking suitable representations to minimize the negative impact of long-tailed imbalanced sample classification [12]. Zhang et al. proposed using an extra buffer as an unbiased dictionary, periodically monitoring the training history corresponding to model updates, and finding meaningful samples from the training data as reward data. The unbiased dictionary continuously updates and provides reward information to optimize class-imbalanced sample weights. This method does not rely on additional generated data, and it can be

directly applied to solving the common long-tail imbalanced sample identification data bias [13].

2.2. Multi-Scale Feature Imbalance in Object Detection

Targeting the problem of the imbalanced scale of object features extracted by object detection backbone network, Liu et al. proposed Path Aggregation Network (PANet), which creates bottom-up path augmentation, shortens information paths, and enhances feature pyramids with accurate localization signals present in lower layers. At the same time, it recovers the damaged information paths between each proposal and all feature levels. This adaptive feature pool is designed to aggregate all feature-level features in each proposal interval [14]. Zhao et al. proposed a multi-level feature pyramid network, which fuses basic features with the largest output feature map of alternately connected, aggregates multi-level, and multi-scale features through an attention mechanism, and finally builds a more effective feature pyramid to detect objects of different scales [15]. Xu et al. proposed the feature pyramid automatic connection structure. Four other scale feature map transformation methods were designed before determining the feature map cascade method of different scales in the backbone network. By traversing the transformation methods of all feature maps, the feature pyramid determines the optimal multi-scale feature fusion method [16]. Kong et al. redefined the feature reconstruction function of the feature pyramid, using global attention to emphasize the global information of the entire image and then performing local reconstruction to local model patches in the receptive field, propagating strong semantics across all feature maps different scales [17]. Kong et al. proposed a reverse connection on the traditional CNN structure, enabling forward features to contain more semantic information. The size difference of adjacent feature maps is eliminated by deconvolution, and features of different scales are generated through reverse connection. After the map, candidate boxes of various sizes are designed for each feature map to cover objects of different sizes and shapes [18]. Kim et al. proposed a parallel feature pyramid network to solve the feature scale imbalance problem. They first adopted Spatial Pyramid Pooling (SPP) to generate wide feature pyramid pools with feature maps of different sizes and then designed a Multi-Scale Context Aggregation (MSCA) module to resize these feature maps to uniform size [19]. Li et al. proposed a zoom-in-and-out network that utilizes multi-scale feature maps at different depths and actively searches for and activates neurons from low-level locations and high-level semantic feature maps. The network splits anchors of different sizes to give the feature map of each scale its own set of features, and the classifier corresponding to each scale to detect the object of a specific scale range [20].

2.3. Positive/Negative Samples Imbalance in Object Detection

Some scholars have carried out some research on the problem of imbalanced positive/negative samples. Ge et al. proposed introducing an enlargement/reduction factor k in the first-stage region proposal network of Faster R-CNN [35], and k adjusts the number of positive/negative samples through competition. This gradient annealing strategy can balance the influence of positive/negative samples [21]. When using anchors to generate proposal regions, because the value of the threshold directly determines the ratio of positive/negative samples, Han et al. proposed using a segmentation strategy to determine the anchor threshold for the balance of positive/negative samples to reduce the impact of negative samples on object detection performance [22]. In the anchor-free target detection task, Hou et al. proposed using hard-positive samples, easy-negative samples, and hard-negative samples to balance the number of positive/negative samples [23]. In the two-stage object detection algorithm, Li et al. proposed increasing the number of positive samples and balancing the proportion of positive/negative samples by reducing the threshold of the anchor and introducing the boundary map of the object's actual value [24]. Li et al. directly introduced the online hard sample mining module into the backbone network of Faster R-CNN, which instantly expanded the number of candidate regions of the object

and avoided the problem of imbalanced positive/negative samples based on anchors [25]. Li et al. proposed a method to embed spatial attention and non-local attention modules in object detection, strengthen more significant regional features, suppress remaining unimportant regional features, and achieve the goal of balancing positive/negative samples [26]. Based on YOLOv3, Li et al. proposed combining self-attention and feature pyramid network, adding the weight of positive samples to the end features of the backbone network, improving the extraction ability of deep features, and alleviating the imbalance between positive/negative samples [27]. Based on SSD, Xu et al. proposed the concept of a core anchor, which regresses conventional multi-scale dense anchors into a square anchor. This proposed core anchor is equivalent to a type of preprocessing, which achieves the purpose of balancing positive and negative samples [28]. Li et al. proposed a new soft edge focus loss function, which introduced a penalty function to expand the center distance between positive/negative samples and solved the problem of difficult object detection caused by the imbalance of positive/negative samples [29]. Based on the focal loss function, Li et al. also considered the number of samples in different classes. They combined the mean square error loss function when regressing the object, further alleviating the positive/negative sample imbalance [30]. Under the framework of YOLOv3 object detection, Li et al. adopted the method of multi-scale clustering anchors, separately clustered according to the size of the output port, and achieved the purpose of balancing positive/negative samples by generating multiple anchors and using focal loss [31]. Based on SSD, Lu et al. proposed an improved confidence loss function by introducing two adjustment factors to adjust the contribution of positive and negative samples [32]. Zheng et al. proposed calculating the ratio of the number of positive and negative samples in the dataset and using the inverse of the ratio as the sample weight to deal with the imbalance of positive/negative samples [33]. Li et al. proposed a weighted binary cross-entropy loss function, which uses the proportion of negative samples in the total samples as the weight to calculate the loss. When the number of samples is imbalanced, the model can better consider the learning of positive samples and obtain better object detection results [34]. Pang et al. proposed a balanced feature pyramid, which firstly adjusted the features of different levels to the same size based on interpolation or max pooling, and then took the mean to obtain the balanced semantic features. Finally, the original features of each layer are enhanced by reversing the operation [36].

2.4. Electronic Component Detection in PCB Assembly Scene

For the electronic component detection problem in the PCB assembly scene, the current research mainly includes six aspects: small size object detection [37,38], PCB positioning [39,40], electronic component detection [41–46], model lightweight [47,48] and real-time detection [49,50]. Li et al. proposed a detection method for small-sized PCB electronic components based on multiple detection heads. This method integrates shallower feature map semantics and location information to improve the detection accuracy of many small-sized objects [37]. Liu et al. proposed a CNN combined with a residual network for object detection of small-sized electronic components [38]. Zhao et al. realized the multi-task detection of category, position, and angle in the PCB assembly process by adding an angle detection task based on the Faster R-CNN [39]. For the precise positioning problem in PCB automatic assembly, Tsai et al. proposed and compared four models: simple multi-layer perceptron (MLP), convolutional neural network (CNN), and CNN combined with support vector regression (SVR). They complete the PCB regression positioning problem quickly and accurately [40]. Lin et al. used the YOLO algorithm to design an automatic positioning and fast identification model for nine kinds of capacitors on the PCB [41]. Kuo et al. designed a three-stage detector to detect electronic components on PCB. In addition to a class-independent region proposal network, the three-stage detector also proposed a graph network to refine the features of electronic components [42]. Lu et al. proposed a method for object detection of scattered electronic components using YOLOv3 [43]. Mukhopadhyay et al. used a color space conversion model to convert

RGB to YCbCr when detecting IC (integrated circuits) on a circuit board. Then, three shape descriptors were combined to achieve the best detection effect [44]. To improve the accuracy of object detection of electronic components on PCB, Liu et al. proposed a new box regression loss function based on YOLOv4, called Gaussian Intersection Joint (GsIoU), which uses the Gaussian function to predict boxes under different anchors. Combined at the same location, the box regression loss is calculated, which ultimately improves the accuracy of the final box regression [45]. Li et al. proposed an anchor allocation strategy based on the effective receptive field in the problem of object detection in the whole scene of PCB assembly, which can improve the accuracy of object detection and reduce the computational complexity [6]. Baranwal et al. used VGG16 to propose a classification model for correctly placing surface mount components during PCB assembly. The classification accuracy was higher than trained artificial vision classification [46]. Shen et al. proposed a PCB defect detection model after assembly. The model consists of a lightweight electronic component detection module and a character recognition module on electronic components. This method is based on Faster R-CNN with context-aware ROI (Region of Interest) pooling and spatial transformer networks to meet industrial needs [48]. Li et al. proposed an object detection model for PCB electronic components based on effective receptive field-anchor matching. By analyzing the influence of different depth backbone networks on the size change of the effective receptive field of the anchor allocation layer, a modular combination strategy of the backbone network is designed to realize the lightweight of the model [47]. Aiming at the problem of visual real-time detection of electronic components on the assembly line by industrial robots in the electronics industry, Guo et al. proposed to use the multi-scale attention module to adaptively fuse the middle and high-level features extracted by Tiny-YOLOv4, which meets the requirements of real-time detection in industrial environments [49]. Shuai et al. proposed a secondary screening detection method to detect electronic components and characters on electronic components in real-time. They first searched and detected the features of electronic components through a gradient boosting decision tree model, and then used CTPN (Connectionist Text Proposal Network) + Tesseract-OCR (Optical Character Recognition) deep learning technology to complete character recognition on electronic components [50].

Although many researchers have made significant progress in the above areas, for the first time we combine solving the imbalance problem that affects the accuracy of object detection and detection efficiency with the multi-class object detection problem in PCB assembly scenarios. Our proposed method will fill some of the knowledge gaps in the above survey methods.

3. Methodology

This study proposes a balanced split of Train/Val, the addition fusion of feature layers, and the efficient anchor concept, which solves the imbalance problem of PCB assembly scenes and improves the accuracy and speed of object detection. The algorithm uses YOLOv3 as the primary network. It takes PCB, electronic components, and through holes as detection objects in two scales of PCB assembly scenarios, and all classes are equally split in the Train/Val. Because the method proposed in this paper solves three imbalanced problems in object detection, for the convenience of the description below, the method proposed in this paper is called Balanced-YOLOv3.

The overall framework of Balanced-YOLOv3 is shown in Figure 1. It mainly includes three parts: the balanced classes split of Train/Val, the multi-scale feature fusion method, and the proposal and acquisition method of the efficient anchor. These three parts occur in the preprocessing before the CNN learning, the connection between the backbone network and the detection head, and the pre-selection box generation stage.

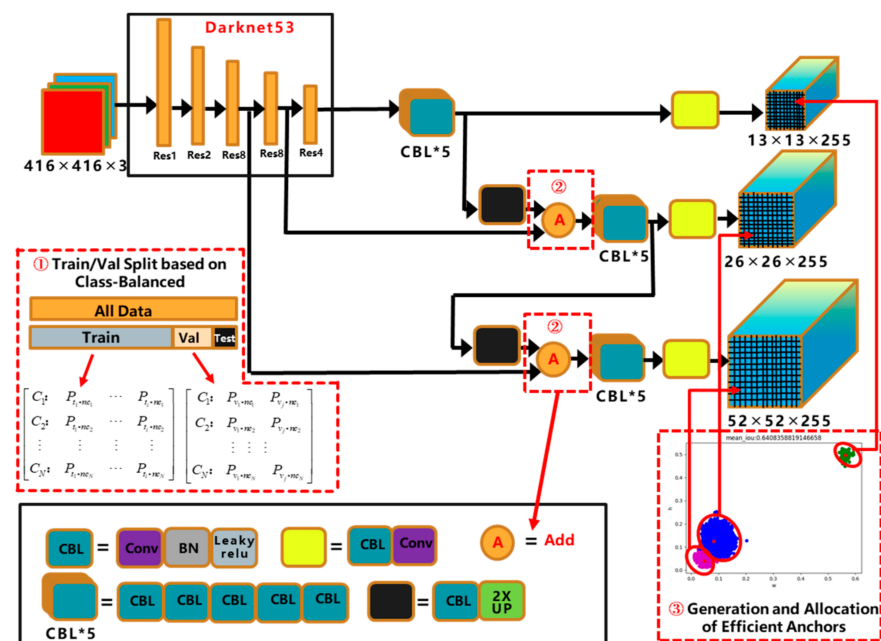


Figure 1. A framework of Balanced-YOLOv3.

3.1. Class-Balanced Train/Val Split

3.1.1. Function of the Training Set, Validation Set, and Test Set

The object detection method based on deep learning is realized by training CNN with big data. In particular, the convolution kernel weight of each layer of the CNN is learned by data-driven learning. We hope that the object detection algorithm obtained has a strong generalization ability. Generalization ability refers to the ability of the trained algorithm to predict unknown data. Our modeling aims to make the model have a better predictive power for known and unknown data. Our modeling seeks to give the algorithm a better predictive ability for known and unknown data.

Before training the CNN object detection algorithm, we need to divide the dataset into the training set, validation set, and test set in advance. The training set is the dataset used to train and make the algorithm learn the features hidden in the data. In each epoch, the same training data is repeatedly fed to the neural network to make the model learn the features of the data. The training set should have a diverse input set so that the model is trained on all scenarios and can predict any unseen data samples that may arise in the future. The validation set is used to validate our model performance during training. This validation process provides information that can help us tune the model's hyperparameters and configuration accordingly, like a trainer telling us whether the training is going in the right direction. The algorithm is trained on the training set, while the model is evaluated on the validation set after each epoch. The main idea of the validation set is to prevent the model from overfitting. The algorithm becomes very good at classifying samples in the training set but fails to generalize and accurately classify data it has never seen before. The test set, training set, and validation set are separated. After the algorithm is trained and validated, the test set is used to evaluate the algorithm's performance objectively. The training, validation, and testing process of the entire dataset in the object detection algorithm is shown in Figure 2.

3.1.2. Train/Val Split Based on Class Balance

Usually, the division ratio of the training set, validation set, and test set is randomly assigned to 60%, 20%, and 20% or 70%, 20%, and 10%, and other combinations according to the number of photos. If the proportion of the training set is too small, the obtained model is likely to be very different from the model obtained from the total data; if the ratio of the training set is too large, the reliability of the test results will be reduced. At the same

time, the number of sample classes in the data set is naturally imbalanced, and the split of the dataset should maintain the consistency of the data distribution as much as possible to avoid the impact on the final result caused by the introduction of additional bias in the data split process. Therefore, after first determining the test set, if the remaining data set is divided into the training set and validation set, the proportion of each class is very different. The error estimation will be biased due to the difference in the distribution of Train/Val. To avoid the problem of large feature learning bias caused by the large difference in the proportion of each class during division, this paper proposes a Train/Val split method based on sample class balance. Figure 3 shows the method proposed in this paper and the conventional method of randomly dividing the Train/Val according to the ratio of the photo's number.

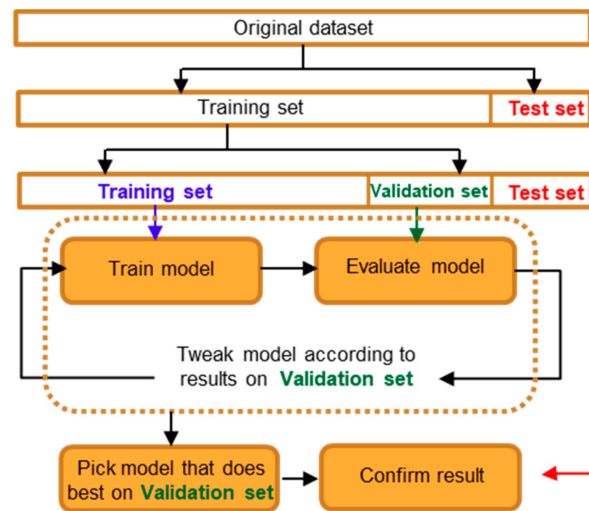


Figure 2. The execution process of the training set, validation set, and test set in object detection algorithm.

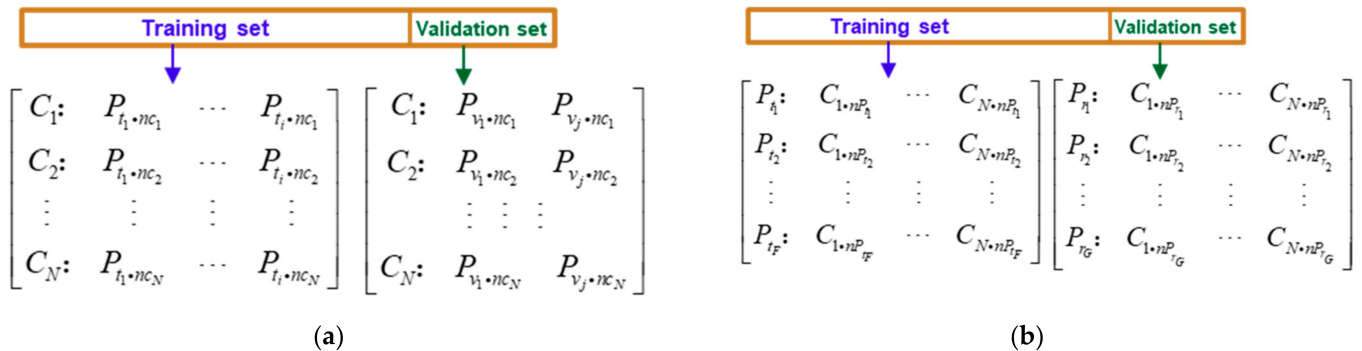


Figure 3. Class-balanced Train/Val split and random Train/Val split according to the number of photos. (a) Train/Val Split based on Class-Balanced. (b) Train/Val Randomly Split based on the Ratio of Photos Number.

In Figure 3a, we use $C_1, C_2 \dots C_N$ to represent N class samples in the dataset, $P_{t_1 \cdot nc_1}, P_{t_1 \cdot nc_2} \dots P_{t_i \cdot nc_N}$ which means the number of samples with classes $C_1, C_2 \dots C_N$ on the i -th photo in the training set. $P_{v_1 \cdot nc_1}, P_{v_1 \cdot nc_2} \dots P_{v_j \cdot nc_N}$ represents the number of samples with classes $C_1, C_2 \dots C_N$ on the j -th image in the validation set. We define $(train_cb)_{C_x}$ the sum of the number of samples with class C_x in the class-balanced training set $(val_cb)_{C_x}$ as the sum of the number of samples with class C_x in the class-balanced validation set. For a class-balanced Train/Val, the following formula holds:

$$(train_cb)_{C_x} = \sum (P_{t_1 \cdot nc_x} + P_{t_2 \cdot nc_x} + \dots + P_{t_i \cdot nc_x}) \quad (1)$$

$$(val_cb)_{C_x} = \sum (P_{v_1 \cdot nc_x} + P_{v_2 \cdot nc_x} + \dots + P_{v_j \cdot nc_x}) \quad (2)$$

$$C_x : (train_cb)_{C_x} / (val_cb)_{C_x} train - val(cb)_radio, x \in [1, \dots, N] \quad (3)$$

In Figure 3b, the result of random training set-validation set division according to the proportion of photos, N represents the total number of classes in the dataset, F represents the total number of photos in the training set, G represents the total number of photos in the validation set. $P_{t_1}, P_{t_2} \dots P_{t_F}$ represents the 1st, 2nd— F -th photo in the training set, $C_{i \cdot nP_{t_1}}, C_{i \cdot nP_{t_2}} \dots C_{i \cdot nP_{t_F}}$ represents the number of samples of the i class in the training set $P_{t_1}, P_{t_2} \dots P_{t_F}, P_{r_1}, P_{r_2} \dots P_{r_G}$ represents the 1st, 2nd—the G photo in the validation set. For the random Train/Val split method based on the proportion of photos, we define $(train_random)_{C_x}$ as the sum of the number of samples with class C_x in the training set and $(val_random)_{C_x}$ as the sum of the number of samples with class C_x in the validation set. The following formula is established:

$$F/G = train - val_radio \text{ of the number of photos} \quad (4)$$

$$(train_random)_{C_x} = \sum (C_{x \cdot nP_{t_1}} + C_{x \cdot nP_{t_2}} + \dots + C_{x \cdot nP_{t_F}}) \quad (5)$$

$$(val_random)_{C_x} = \sum (C_{x \cdot nP_{r_1}} + C_{x \cdot nP_{r_2}} + \dots + C_{x \cdot nP_{r_G}}) \quad (6)$$

$$C_x : (train_random)_{C_x} / (val_random)_{C_x} = random \text{ number}, \quad (7)$$

The above analysis shows that, in general, the number of samples of different classes in the entire dataset present a long-tailed class imbalance. The ratio of the numbers in the training set and the validation set must be random, and the ratio of different classes is very different, which will inevitably lead to the problem of large model learning deviation and reduce the detection performance of the model. However, in the Train/Val split result based on the balance of sample classes, the ratio of the number of training set samples to the number of the validation set samples for all classes is a fixed value. The split result can maintain the Train/Val class balance. The consistency of the data distribution avoids the introduction of additional bias due to the large difference in the proportion of each class. Finally, an object detection model with superior performance can be trained. Algorithm 1 gives the Train/Val split method based on the balance of sample classes and shows how the data set is divided into the Train/Val. The whole process does not contain the test set.

Algorithm 1: Train/Val Split based on Class-Balanced

Input: All photos and XML files containing label information for all class samples.

Output: Photos and XML files of the training set. Photos and XML files of the validation set.

1: Create a list<map> corresponding to each XML file, including the name of the xml file and the number of each class in the XML.

```
[[
    "name": "xxx.xml",
    "C1": number1,
    "C2": number2,
    ———
    "CN": numberN,
  ]]
```

2: shuffle list<map>

3: Sum the number of classes in all list<map>

4: Find the class with the smallest sum: objectMinName

5: Pick xml files and photos that contain objectMinName, List<objectMinName>

6: Sort List<objectMinName> in descending order

7: Split according to a predefined ratio of class balance List<objectMinName> => train_List: val_List

8: Aggregate zongtrain_List: zongval_List

9: Repeat the class number times of step3–step8

10: Get the final zongtrain_List: zongval_List

11: end

3.2. An Addition Fusion Strategy for Multi-Scale Imbalanced Features

As an emerging feature extractor, CNN's forward convolution, pooling, and other processes are the feature extraction process, especially the image features in the visual task can be extracted. At the terminal of CNN, the mapping function of the task will be linked, and the feature will be mapped to the predicted results. The back propagation process is to update the weight parameters of CNN according to the chain derivation rule. There are two main reasons for feature fusion as an essential means to achieve multi-scale object detection in CNN. First, the receptive fields of feature maps of different sizes are different. When the image undergoes the convolution operation, the feature map will become smaller as the convolution deepens. In reducing the feature map, each pixel combines the information of multiple feature points in the original feature map. That is, the receptive field will become larger. To detect objects of different sizes, we need to fuse the receptive fields of various sizes. Second, the feature maps of different depths contain different information. Low-level features have higher resolution and contain more location and detail information, but they have lower semantics and more noise due to less convolution. High-level features have stronger semantic information, but the resolution is low, and the perception of details is poor. The problem we face is how to make the two better complement each other to achieve the best detection effect.

3.2.1. Concatenation and Addition Feature Fusion Methods

The commonly used feature fusion methods mainly include concatenation and addition. The concatenation operation links together feature maps with different channel numbers. That is to say, the features describing the image itself are increased, but the information under each feature is not improved. The addition operation is the superposition between the same level feature map information, and the number of channels does not change. The addition means that the amount of information describing the feature map of the image increases, but the dimension representing the image itself does not increase, but the amount of information in each dimension increases. Whether it is concatenation or addition, the deep network contains the information of the shallow network, so it can reach the shallow network through different branches during backpropagation, increasing the generalization ability and feature expression ability of the network. The most apparent difference between concatenation and addition is the change in data dimension. The addition operation requires that the data dimensions are precisely the same, and the added dimension is the same as before. The concatenation operation requires that the data have a different dimension, generally the number of channels. The dimension, height, and width of the image are the same, and the number of channels after merging is the sum of the number of data channels participating in the merging.

Suppose $X_i \in R^{H_i \times W_i \times C_i}$ ($i \in \{1, 2, \dots, L\}$) and $Y_j \in R^{H_j \times W_j \times C_j}$ ($j \in \{1, 2, \dots, J\}$) are the two sets of inputs before feature fusion, and each feature map has N objects. Taking the X_i feature map as an example, the feature map contains the semantic information of all objects $X_{i,s} = s_1 + s_2 + \dots + s_n$ ($n \in \{1, 2, \dots, N\}$) and the location information of all objects as $X_{i,l} = l_1 + l_2 + \dots + l_n$ ($n \in \{1, 2, \dots, N\}$). W_i and W_j are the convolution kernels of the corresponding channels. The symbol $*$ represents the convolution operation. The formulas for defining concatenation and addition, respectively, are shown in Equations (8) and (9):

$$Z_{l.concat} = \{X_i * W_i, Y_j * W_j\}, l \in \{1, 2, \dots, L + J\} \quad (8)$$

$$Z_{l.add} = \{(X_i + Y_i) * W_i\}, l \in \{1, 2, \dots, L\} \quad (9)$$

3.2.2. Invariance of Semantic Information

The so-called invariance means that for a function, if the transformation you apply to it does not affect the output at all, then the function is invariant to the transformation.

Assuming that the input is x and the function is f , if we first transform the input g , $g(x) = x'$ if there is

$$f(x) = f(x') = f(g(x)) \quad (10)$$

Then the f is said to be invariant to the transformation g .

In the convolutional neural network, any feature map of the two inputs X_i and Y_j before feature fusion comes from the original image. If there are N objects in the original image, then the semantic information of any feature map in X_i and Y_j can be expressed as:

$$Z_i(s) = s_1 + s_2 + \dots + s_n (n \in \{1, 2 \dots N\}) \quad (11)$$

For the feature fusion method of concatenation, because it is only a stack of input feature maps, the semantic information of any feature map after concatenation can be expressed as:

$$Z_{i\text{-concat}}(s) = s_1 + s_2 + \dots + s_n (n \in \{1, 2 \dots N\}) \quad (12)$$

$$Z_i(s) = Z_{i\text{-concat}}(s) \quad (13)$$

The feature fusion method of addition is the superposition of the corresponding input feature maps. The feature map obtained after superposition does not increase the number of objects in the original image and change the class, so the semantic information of the feature map after addition can be expressed as:

$$Z_{i\text{-add}}(s) = s_1 + s_2 + \dots + s_n (n \in \{1, 2 \dots N\}) \quad (14)$$

$$Z_i(s) = Z_{i\text{-add}}(s) \quad (15)$$

According to Equations (13) and (15), we can find that the concatenation and addition transformations applied to the semantic information of the input feature map will not affect the original semantic information at all. Therefore, whether it is concatenation or addition, these two feature fusion methods can maintain the invariance of semantic information.

3.2.3. Equivariance of Location Information

The so-called equivariance means that for a function, if the transformation you apply to its input will also reflect on the output, then the function is equivariant to the transformation. That is, for function f and transformation g , if there is

$$f(g(x)) = g(f(x)) \quad (16)$$

Then it is said that f is equivariant to the transformation g .

In the CNN, the position information of any one of the two input X_i and Y_j before feature fusion can be expressed as:

$$Z_{i\text{-concat}}(l) = l_1 + l_2 + \dots + l_n (n \in \{1, 2 \dots N\}) \quad (17)$$

For the feature fusion method of concatenation, the two input features are first concatenated. Then the position information of any feature map after feature fusion is extracted, which can be expressed as:

$$Z_{i\text{-concat}}(l) = l_1 + l_2 + \dots + l_n (n \in \{1, 2 \dots N\}) \quad (18)$$

First, extract the position information of any one of the input feature maps of the two inputs. The location information after concatenating the feature maps from which location information is extracted can be expressed as:

$$\text{concat}(Z_i(l)) = l_1 + l_2 + \dots + l_n (n \in \{1, 2 \dots N\}) \quad (19)$$

$$Z_{i\text{-concat}}(l) = \text{concat}(Z_i(l)) \quad (20)$$

For the feature fusion method of addition, first, add the two input features correspondingly, and then extract the position information of any feature map after the features are added, which can be expressed as:

$$Z_{i-add}(l) = l_1 + l_2 + \dots + l_n (n \in \{1, 2 \dots N\}) \quad (21)$$

First, extract the position information of any one of the input feature maps of the two inputs. Then, add the corresponding positions to the feature maps of the extracted position information. The position information can be expressed as follows:

$$add(Z_i(l)) = l_1 + l_2 + \dots + l_n (n \in \{1, 2 \dots N\}) \quad (22)$$

$$Z_{i-add}(l) = add(Z_i(l)) \quad (23)$$

According to Equations (20) and (23), the concatenation or addition transformation applied to the position information of the input feature map is equivalent to the position information extracted after concatenating or adding the input feature map first. Therefore, whether it is concatenation or addition, these two feature fusion methods can maintain the equivariance of position information.

3.2.4. YOLOv3 Based on Addition Feature Fusion Strategy

Although the two feature fusion methods of concatenation and addition can maintain the invariance of semantic information and the equivariance of position information, the concatenation operation saves important detailed information by increasing the number of channels. The calculation amount of addition is much smaller than that of concatenation, which holds more parameters and calculation amount. At the same time, for small-sized objects, the deeper the layers of the CNN, the smaller the feature map, and the less semantic and location information is retained. When fused with the feature map of the shallow network, the addition operation can strengthen the object semantics and location information. In contrast, the concatenation operation cannot make up for the gradually disappearing target information of the deep network. Therefore, for the problem of small size in object detection in assembly scenes, this paper designs an addition fusion strategy based on multi-scale imbalanced features of the backbone network, as shown in Figure 4:

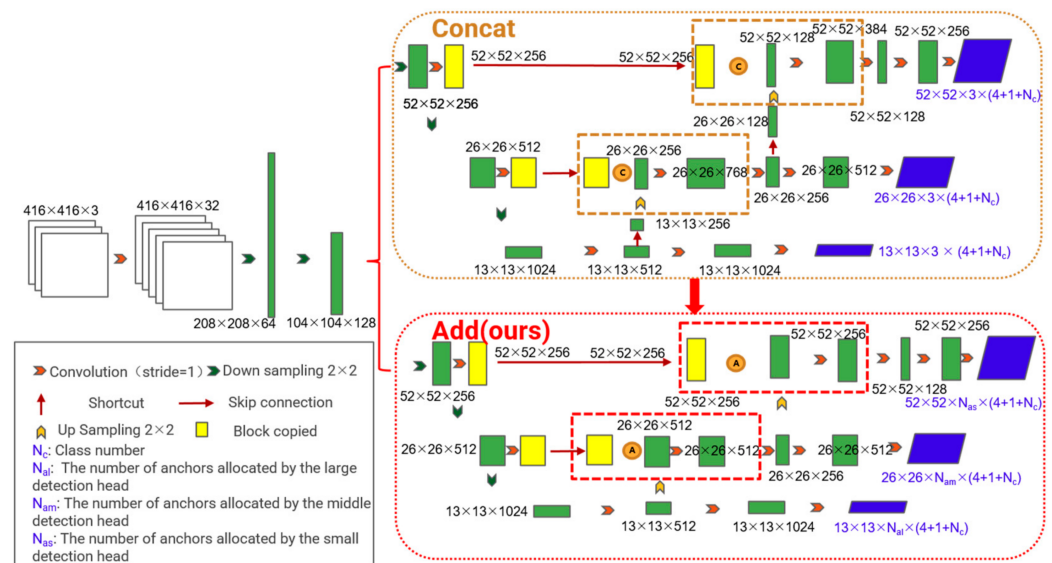


Figure 4. Addition fusion strategy for multi-scale imbalanced features based on YOLOv3.

To further verify the effect of the addition fusion strategy in object detection in PCB assembly scenes, this paper also changed the concatenation fusion in YOLOv3-SPP to addition fusion, as shown in Figure 5:

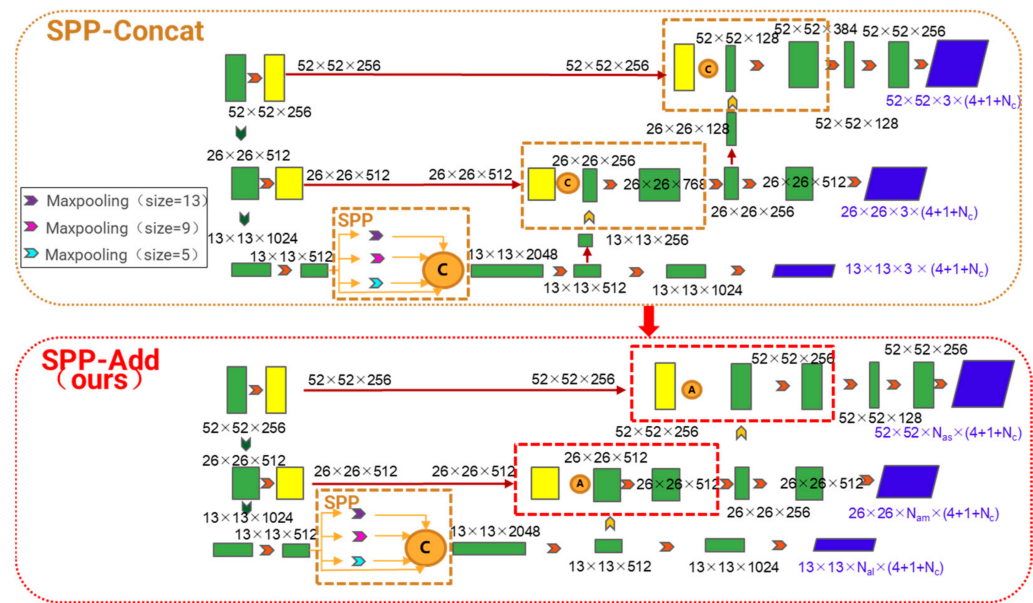


Figure 5. Addition fusion strategy for multi-scale imbalanced features based on YOLOv3-SPP.

Whether it is Figure 4 or Figure 5, we can observe that in the concatenated feature fusion mode of YOLOv3 and YOLOv3-SPP, the $52 \times 52 \times 256$ in the shallow layer, and the up-sampled $52 \times 52 \times 128$ in the middle layer are concatenated after stacking, the output is $52 \times 52 \times 384$, and the number of channels becomes 384. After the $26 \times 26 \times 512$ in the middle layer and the up-sampled $26 \times 26 \times 256$ in the deep layer are concatenated and stacked, the output is $26 \times 26 \times 768$, and the number of channels becomes 768. In the addition feature fusion mode, the corresponding feature maps of $52 \times 52 \times 256$ in the shallow layer and the up-sampled $52 \times 52 \times 256$ in the middle layer are superimposed, and the output is $52 \times 52 \times 256$, and the number of channels is 256. After the feature map $26 \times 26 \times 512$ of the middle layer and the up-sampled feature map $26 \times 26 \times 512$ of the deep layer are superimposed, the output is $26 \times 26 \times 512$, and the number of channels becomes 512. The addition fusion strategy can speed up the learning process of CNN parameters and reduce the amount of calculation by lowering the dimension of the channel. The point-by-point addition operation of the feature map increases the information content of the feature map. It makes up for the information loss defect of small-sized targets caused by the deepening of the network layers.

3.3. Efficient Anchor Design to Alleviate the Imbalance of Positive/Negative Samples

Current anchor-based object detection relies heavily on learning from a large sample of data. Most detectors place many anchors at each pixel on the image or a partitioned dense network and then calculate the IoU (Intersection over Union) between each anchor and its adjacent ground truth. The anchor is a positive sample if the value is higher than the pre-set threshold. Otherwise, the anchor is a negative sample. Then the entire CNN continuously learns the features of positive/negative samples. Finally, the bounding box regression and loss function are used to learn network parameters. If there is a severe imbalance between positive/negative samples, the learning of network parameters will be severely skewed. CNN ignores the feature information of positive samples and over-learns the bias information of negative samples, which reduces the network learning efficiency and improves the detection error rate. Beginning with Faster R-CNN, researchers have improved detection performance by increasing the number of anchors. Many dense anchors of different scales and aspect ratios are placed in the center of a receptive field. Although the use of dense anchors can improve the detection effect of some dense objects, it brings three problems simultaneously. One is that too many anchors will affect the speed of model training. Second, because the anchors are too dense, most of the anchors are distributed

in the background area, resulting in too many negative samples, a serious imbalance of positive/negative samples, and can't play a positive role in the loss function of the object bounding box regression. Third, the allocation of dense anchors often adopts a rough and straightforward average allocation method, which does not consider the difference between the actual sample size and the allocated anchor size, resulting in the problem of limited detection accuracy improvement.

In this paper, the concept of the efficient anchor is proposed by calculating the distribution of the ground truth labels in the dataset at three scales, using two K-means clustering. Efficient anchors get the total number of anchors adapted to the target size and distribution of the dataset. While obtaining the number of effective anchors, the effective allocation of anchors in the three output ports of the detection head is realized. Anchors are a certain number of boxes with different aspect ratios drawn at the center of each pixel. These boxes are not real but preset boxes that the model artificially sets for prediction. Under the idea of dense anchor prediction, the number of anchors is often determined by empirical values, such as 9, 5, 4, 6, etc. The concept of an efficient anchor essentially refers to an anchor that plays an actual role. In the object detection dataset, we used the sum of the maximum times that each object's center of the ground truth appears in the grid corresponding to the output detection port. The preset box generated by clustering is the efficient anchor. The generation and allocation method of efficient anchors proposed in this paper is shown in Figure 6. As shown in Figure 6, the generation and allocation of effective anchors are mainly composed of two k-means clustering in the dataset and the statistics of the number of object mapping regions. The k of the first k-means clustering is determined by the output number of the YOLOv3 detection head because the YOLOv3 detection head has three scale outputs, so the first clustering produces three regions. These three regions are arranged from small to large, corresponding to sizes 13×13 , 26×26 , and 52×52 , respectively. One statistic is that when each grid of the three output ports is mapped back to the original image, we count the maximum number of center points of the object ground truth in the dataset that falls within the grid of the three output ports. The second clustering is to cluster the corresponding numbers of the three regions according to the respective objects of the three output ports. The generation and allocation steps of the whole efficient anchor are as follows:

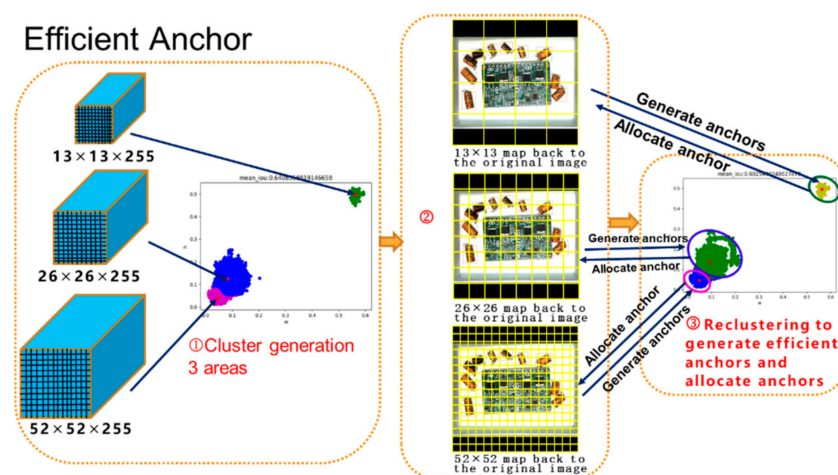


Figure 6. Generation and allocation of efficient anchors.

Step 1: Load the label data of all objects in the training set, and normalize the width, height, and center point of the label data according to the size of the image where the object is located;

Step 2: It is known that the YOLOv3 detection head has three output ports, and the K-means clustering algorithm is used to cluster the normalized label data of all objects to generate three sets of object data;

Step 3: Sort the cluster center point sizes of the three data groups. The group of data to which the largest size belongs corresponds to the 13×13 output port, the group of data to which the middle size belongs corresponds to the 26×26 output port, and the group of data to which the smallest size belongs corresponds to 52×52 output port;

Step 4: Resize all original images in the training set to 416×416 , and adjust all object sizes on the original image in equal proportions. For the three output ports 13×13 , 26×26 , and 52×52 divide the adjusted images respectively 169 grids of 32×32 , 676 grids of 16×16 , and 2704 grids of 8×8 ;

Step 5: For the group of sample labels with the largest size in Step 3, corresponding to the grid of size 32×32 , the center points of the ground truth of these samples are randomly distributed in various positions of the picture. We count the number of sample center points that fall into each grid and take the maximum number of center points that appear in each grid n_{al} as the number of assigned anchors for this output port;

Step 6: According to Step 5, count the number of anchors n_{am} and n_{as} assign them to the output port corresponding to the two groups of data of medium size and minimum size;

Step 7: Perform the second K-means clustering on the three data sets in Step 2. The statistics n_{al} , n_{am} and n_{as} determined the number of clusters of the three data sets, and the three sets of data are clustered. The result is the final effective anchor;

Step 8: The anchors clustered from the three sets of data are the allocated anchors of the corresponding output ports.

The generation and allocation of efficient anchors are obtained based on the statistical analysis of the historical data of the training set samples. By summarizing the spatial location distribution information of the samples, the anchors that are suitable for the scene and work are generated. Compared with dense anchors, efficient anchors are small in number and contain the output distribution information of most detection objects. While improving the preset effect of anchors, the proportion of positive samples corresponding to anchors is increased, and the proportion of negative samples is increased and reduced to alleviate the problem of positive/negative sample imbalance.

4. Experimental Verification and Analysis

We evaluate the proposed method on the PCB assembly scene object detection dataset. The dataset contains two images of size 818×600 and 4092×3000 , with 21 detection object classes. We will first introduce the experimental comparison effect of the three innovation points proposed in Section 3 and then use the subjective and objective evaluation indicators to comprehensively evaluate the Balanced-YOLOv3 proposed in this paper. Then we will compare Balanced-YOLOv3 with other current object detection algorithms for detection accuracy and computational complexity, and finally, we will analyze and discuss the method in this paper.

4.1. Class-Balanced and Random Photo-Number Ratio Train/Val Split Results

The object detection dataset of PCB assembly scene is a self-built dataset that simulates the whole scene of PCB through-hole electronic component assembly, including three scenarios before assembly, during assembly, and after assembly, involving PCB, some SMD electronic components on PCB, through-holes of inserted and to-be-inserted, and electronic components before and after assembly. There are 21 classes of objects, with 1000 photos and 9725 objects.

We first randomly divided the test set from the whole dataset for the entire PCB assembly scene dataset according to 10% of the photos. The remaining data was used as a combination of Train/Val. Then, according to the class-balanced Train/Val split method and the image number ratio Train/Val random split method, the entire dataset is split into cb_82 and random datasets. The number of different classes in the test set and the ratio of Train / Val in the two datasets are shown in Figure 7:

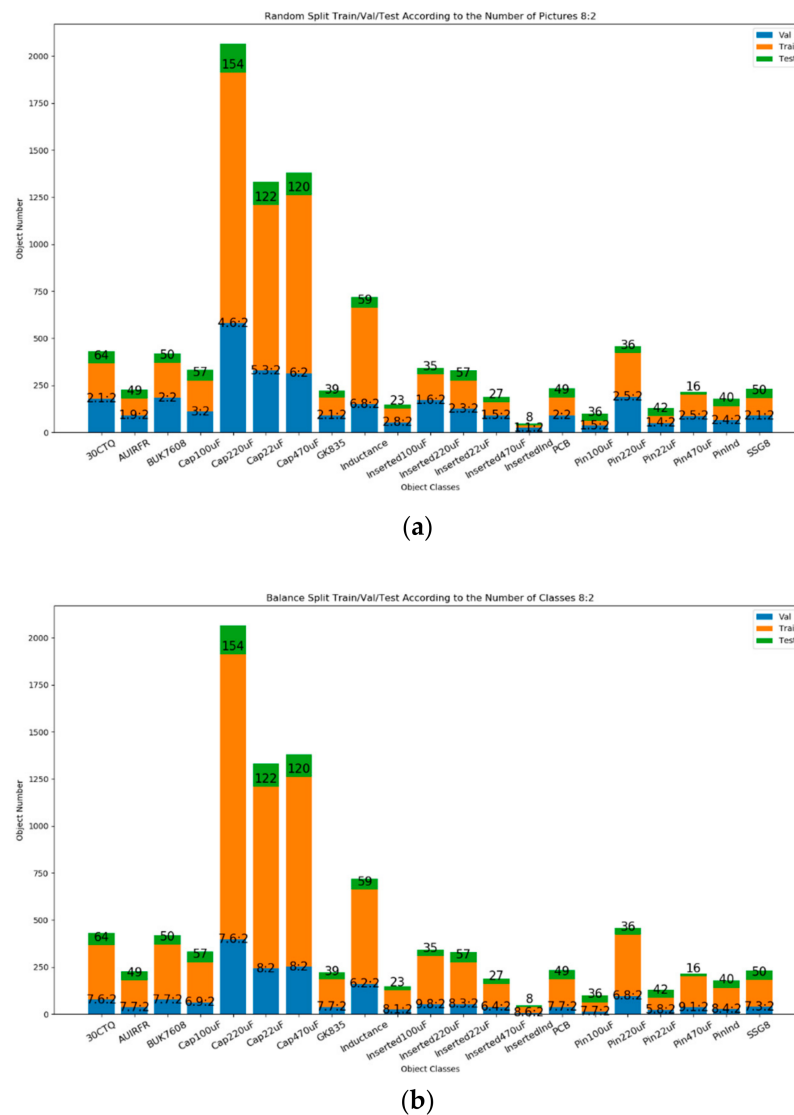


Figure 7. Statistics chart of the number of classes in two ways of splitting the dataset. (a) Randomly split Train/Val (train_picture_number:val_picture_number = 8:2). (b) Class-balanced Train/Val split (train_calss_number:val_calss_number = 8:2).

In the stacked bar chart in Figure 7, green, yellow, and blue represent the distribution of the 21 classes in the test set, training set, and validation set, respectively. The height of the total column of each class represents the total number of objects in that class. From Figure 7, we can see that the number of different classes is highly imbalanced. The numbers on the green column represent this class's specific number of objects in the test set. The numbers in the yellow and blue areas represent the ratio of objects of this class in the Train/Val under the two split methods. It can be seen from Figure 7a that in the Train /Val randomly split according to the ratio of the picture's number of 8:2, the corresponding ratios of different classes are also random. Figure 7b is the Train/Val split according to the class-balanced of 8:2. The corresponding proportions of other classes in the figure are close to 8:2. We name the dataset corresponding to Figure 7a as a random dataset and the dataset corresponding to Figure 7b as the cb_82 dataset. Section 4.4 will verify in experiments that the model trained on the cb_82 dataset has better object detection performance than the random dataset for the same test set.

4.2. Contrastive Analysis of Concatenation Feature Fusion and Addition Feature Fusion

The objects in the object detection dataset of the PCB assembly scene have imbalanced scale distribution. Figure 8 shows the area ratio of all samples in the dataset to their pictures. The symbol 'X' in Figure 8 represents the ratio of the area of each object to the total area of the image where the object is located, and the blue error bar represents the size of the error in the distribution of the area ratio class. The longer the error bar, the more dispersed the distribution of the class area ratio. The shorter the error bar, the more concentrated the distribution of the class area ratio. From Figure 8, we can see that an object such as PCB occupies a large area, accounting for about 25–35% of the entire picture area. The area ratios of the other 20 classes of objects are all below 10%, indicating that the object size distribution of the dataset is imbalanced, and most of the objects are small-sized objects.

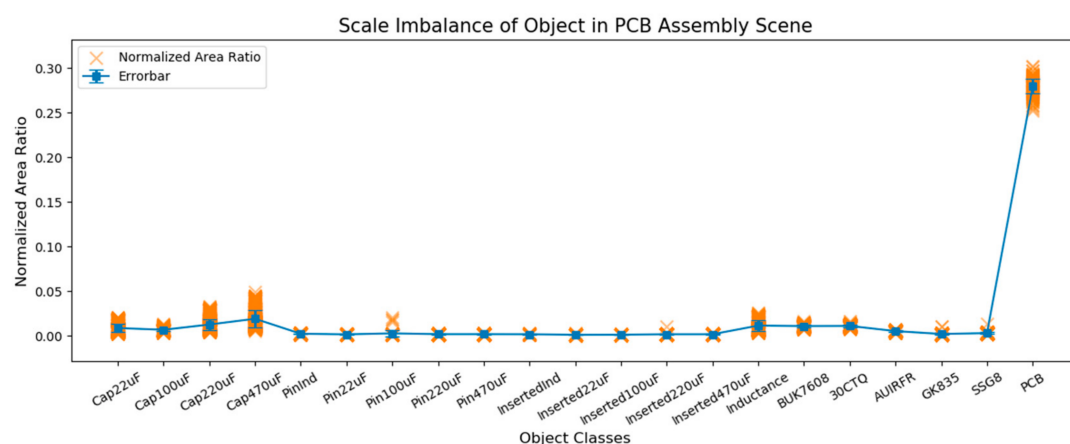


Figure 8. Statistics chart of the normalized area ratio of all objects.

After the object with imbalanced size distribution and small size is extracted through the backbone network at multiple scales, there is the problem of imbalance between shallow and deep features and the phenomenon of feature disappearance in the deep network. We represent ③ and ④ for the concatenation and addition strategies in feature fusion, respectively. Figure 9 shows the results of two feature fusions under the four algorithms of YOLOv3-concat, YOLOv3-add, YOLOv3-spp-concat, and YOLOv3-spp-add after adopting concatenation and addition, respectively.

Figure 9a shows two apparent objects in the input picture. Figure 9b,d,f,h shows the feature fusion effect diagrams of the deep backbone and the intermediate layer network after up-sampling in the four models. Figure 9c,e,g,i shows the feature fusion effect diagrams after the up-sampling of the backbone intermediate layer network and the shallower network in the four models. Figure 9b,f shows whether the input image is passed through YOLOv3 or YOLOv3-spp. Even if the deep network features are concatenated, part of the semantics information will be lost. The corresponding Figure 9d,h belongs to the addition fusion of the deep network. It can be seen that the fused feature map retains the semantics information of two objects in the original image. Figure 9c,g shows that after the input image passes through YOLOv3 and YOLOv3-spp, the concatenation fusion feature maps of the middle layer and the shallower layer retain some position information of the two objects. The corresponding Figure 9e,i belong to the addition fusion of the intermediate layer and the shallower network. In addition, it can be seen that the feature map retains the two objects in the original image. The contour and detail features make recognizing and locating the object easier. Compared with the concatenation, the addition can solve the problem of losing object semantic and position information caused by the imbalance of multi-scale features to improve the detection effect. It can reduce the number of channels after fusion, reduce the number of model parameters, and improve the detection speed.

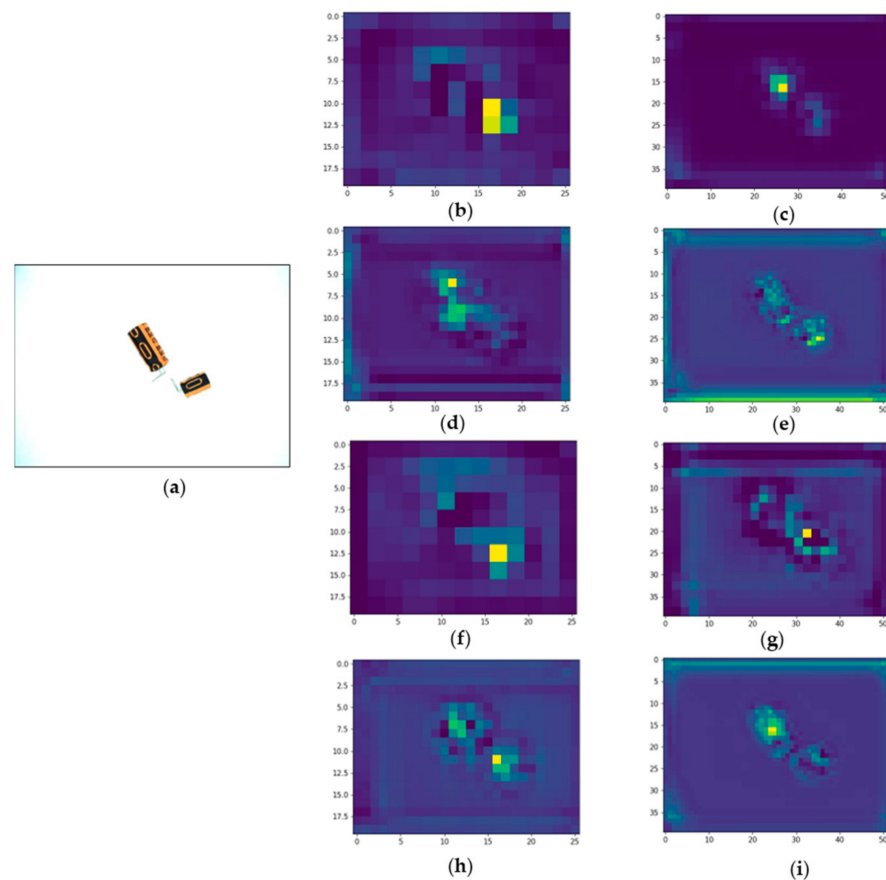


Figure 9. Concatenation and addition feature fusion effect diagram based on YOLOv3 (a) Input Picture. (b) YOLOv3 first-level concatenation output feature map (YOLOv3:26 × 26 × 512 © 26 × 26 × 256 = 26 × 26 × 768). (c) YOLOv3 second-level concatenation output feature map (YOLOv3:52 × 52 × 256 © 52 × 52 × 128 = 52 × 52 × 384). (d) YOLOv3-add first-level addition output feature map (YOLOv3-add:26 × 26 × 512 (A) 26 × 26 × 512 = 26 × 26 × 512). (e) YOLOv3-add second-level addition output feature map (YOLOv3-add:52 × 52 × 256 (A) 52 × 52 × 256 = 52 × 52 × 256). (f) YOLOv3-spp first-level concatenation output feature map (YOLOv3-spp:26 × 26 × 512 © 26 × 26 × 256 = 26 × 26 × 768). (g) YOLOv3-spp second-level concatenation output feature map (YOLOv3-spp:52 × 52 × 256 © 52 × 52 × 128 = 52 × 52 × 384). (h) YOLOv3-spp-add first-level addition output feature map (YOLOv3-spp-add:26 × 26 × 512 (A) 26 × 26 × 512 = 26 × 26 × 512). (i) YOLOv3-spp-add second-level addition output feature map (YOLOv3-spp-add:52 × 52 × 256 (A) 52 × 52 × 256 = 52 × 52 × 256).

4.3. Comparative Analysis of Dense Anchors and Efficient Anchors

In the traditional YOLOv3 algorithm, to solve the object's multi-scale problem, the K-means clustering method is directly used to generate nine anchors without considering the spatial position distribution of the object in the dataset. For the three scale outputs of the detection head, each output is equally allocated three anchors as a preset box, and the grids of the three outputs are mapped back to the original image. The algorithm achieved training and prediction by densely placing nine anchors in the center of each grid of the original image. We call the traditional YOLOv3 method of generating and allocating anchors dense anchors. The efficient anchor concept proposed in this paper fully absorbs the historical information of the spatial distribution and size of objects in the dataset. The number of efficient anchors is small, which can alleviate the imbalance of positive/negative samples. The allocation of anchors at the corresponding scale is more targeted to the scale object. In Figure 10, the comparison results for the generation and allocation of dense anchors and efficient anchors for the object detection dataset for PCB assembly scenes.

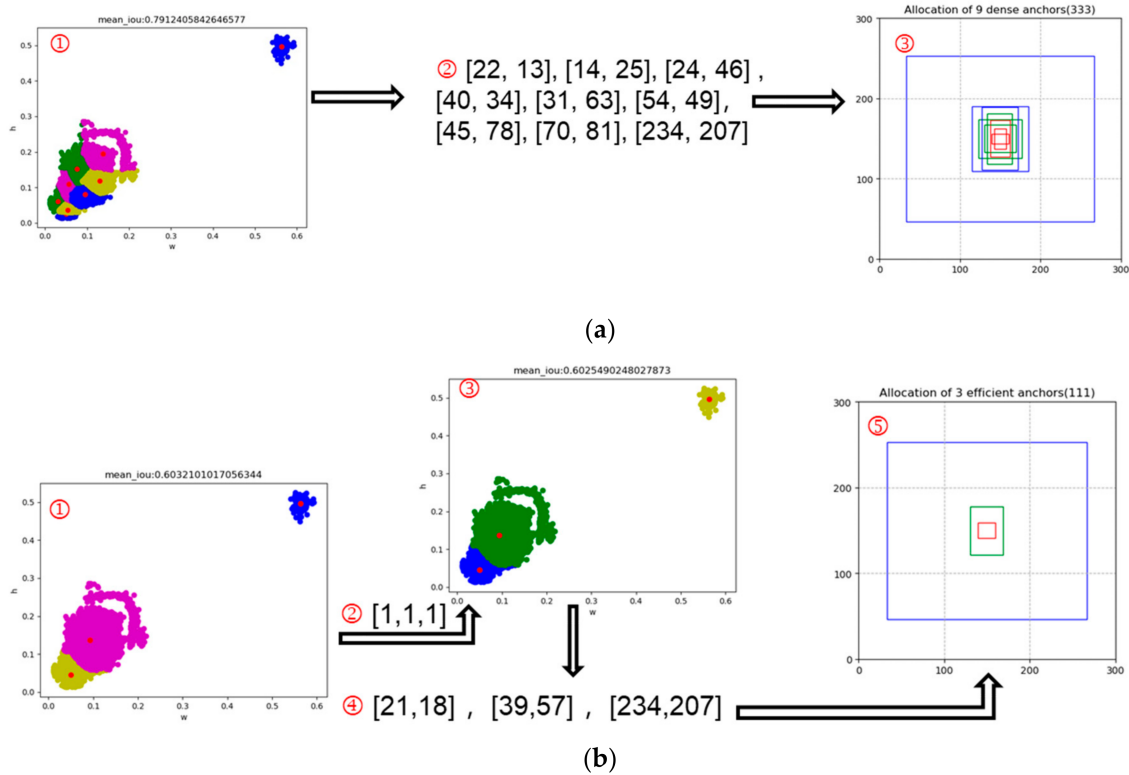


Figure 10. Comparative analysis of the generation and allocation of 9 dense anchors and efficient anchors. (a) Generation and allocation of 9 dense anchors in original YOLOv3. (b) Generation and allocation of efficient anchors.

Figure 10a is a schematic diagram of the generation and allocation of anchors in traditional YOLOv3. The whole process is divided into three steps. ① Divide the width and height of all objects in the training set by the width and height of the image to normalize and use the K-means clustering algorithm to generate nine anchors. ② Arrange the nine anchors generated by clustering from small to large. ③ The nine anchors are equally divided into three groups, blue, green, and red, representing the anchors allocated to the 13×13 , 26×26 , and 52×52 outlets, respectively.

Figure 10b is a schematic diagram of the generation and allocation of efficient anchors. ① Because the detection head has three output ports, the K-means clustering method divides the normalized objects into three regions. ② Map each grid of the three output ports back to the original image and count the maximum number of object centers in the three areas that fall in the three sizes of grids as [1,1,1]. ③ Use the K-means clustering method to cluster the data of each area to generate the corresponding number of anchors. ④ Generate efficient anchors. ⑤ The statistical result is the number of anchors that should be allocated to each output port. Blue, green, and red represent the efficient anchors allocated to the 13×13 , 26×26 , and 52×52 output ports.

Once the number of anchors for each output port is determined, the proportion of positive/negative samples for the entire object detection is also determined. For YOLOv3, we use ps_n to represent the number of positive samples, ns_n to represent the number of negative samples, p_n to represent the number of photos, and a_{pn} to represent the number of anchors on a photo. $IoU(anchor, groud_truth)$ is the intersection over the union of anchor and object ground truth, IoU_t is the threshold for distinguishing positive and negative samples.

$$IoU(anchor, groud_truth) = \frac{|anchor \cap groud_truth|}{|anchor \cup groud_truth|} \quad (24)$$

We count the intersection over the union results between all anchors and objects ground truth in the dataset. As long as the result is greater than or equal to the threshold for distinguishing positive and negative samples, the anchor is positive. Otherwise, the anchor is a negative sample.

$$ps_n = \text{number}[IoU(anchor, groud_truth) \geq IoU_t] \quad (25)$$

For a photo, the number of anchors assigned to the three output ports is known to be n_{al} , n_{am} , and n_{as} , respectively, then the total number of anchors for this photo is:

$$a_{pn} = 13 \times 13 \times n_{al} + 26 \times 26 \times n_{am} + 52 \times 52 \times n_{as} \quad (26)$$

The number of negative samples in the entire dataset is:

$$ns_n = a_{pn} \times p_n - ps_n \quad (27)$$

Using the above determination method of positive/negative samples, for the PCB assembly object detection dataset, we calculated the $IoU(anchor, groud_truth)$ based on nine dense anchors and three effective anchors of all objects, then we set $IoU_t = 0.20$. The positive/negative samples of dense and efficient anchors are counted in Table 1.

Table 1. Positive/negative sample statistics table for dense anchors and effective anchors.

	9 Dense Anchors	3 Efficient Anchors
The total number of anchors	10,647,000	3,549,000
ps_n	88,079	39,894
ns_n	10,558,921	3,509,106
$\frac{ps_n}{(ps_n + ns_n)}$	0.8273%	1.1241%
$\frac{ns_n}{(ps_n + ns_n)}$	99.1727%	98.8759%

From Table 1, in terms of absolute quantity, the number of positive samples generated by nine dense anchors is 2.2 times the number of positive samples generated by three efficient anchors, and the number of negative samples generated by nine dense anchors is three times the number of negative samples generated by three efficient anchors. From the ratio of positive and negative samples to the sum of samples, compared with nine dense anchors with three efficient anchors. However, the number of anchors is reduced, the proportion of positive samples increases, and the proportion of negative samples decreases. From the above analysis, it can be concluded that the efficient anchors designed in this paper can not only realize the generation and distribution of anchors at the same time, alleviate the problem of imbalance of positive/negative samples, but also reduce the number of algorithm parameters and speed up the training process of the algorithm.

4.4. Balanced-YOLOv3 Algorithm Testing and Evaluation

4.4.1. Experimental Platform and Parameter Settings

The algorithm designed in this paper is tested on a deep learning workstation with Intel® Xeon® Gold 6132 CPU @ 2.60 GHz dual processor and 192 GB memory, the graphics card is NVIDIA® Titan RTX with 24 G graphics ram size, the operating system is Ubuntu 18.04 LTS, the program development tool is Python3.7, and the machine learning function library is PyTorch1.10. The baseline of YOLOv3 is written in the Ultralytics PyTorch framework. Table 2 shows the setup parameters for the algorithm experiments:

Table 2. Parameter settings for algorithm experiments.

Parameter	Value
Train image size in pixels (height \times width)	416 \times 416
Number of categories	21
Training epochs	200
Train warmup epochs	3
lr0 (initial learning rate)	0.01
lrf (final learning rate)	0.1
SGD momentum	0.8
iou_t	0.2
Train batch size	16

To better illustrate the effectiveness of Balanced-YOLOv3, we use the concatenation and addition strategies based on YOLOv3 and YOLOv3-spp. Sixteen sets of algorithm experiments are conducted using dense and efficient anchors for randomly split and class-balanced split datasets. Table 3 gives the names of these 16 groups of algorithms and the details of specific method combinations.

Table 3. Algorithm naming statistics for the two Train/Val split datasets and proposed methods.

Algorithm Name	Train/Val Split	Feature Fusion	52 \times 52	26 \times 26	13 \times 13
YOLOv3-9-random	Train/Val Randomly Split based on the Ratio of Photos Number (8:2)	concatenation	[22, 13]	[40, 34]	[47, 78]
YOLOv3-spp-9-random			[14, 25]	[32, 62]	[70, 83]
YOLOv3-sppadd-9-random		add	[24, 49]	[55, 49]	[233, 205]
YOLOv3-add-9-random					
YOLOv3-3-random		concatenation			
YOLOv3-spp-3-random			[22, 23]	[42, 60]	[233, 205]
YOLOv3-sppadd-3-random	Train/Val Split based on Class-Balanced (8:2)	add			
YOLOv3-add-3-random					
YOLOv3-9-cb82		concatenation	[20, 12]	[24, 49]	[55, 50]
YOLOv3-spp-9-cb82			[14, 25]	[40, 34]	[58, 81]
YOLOv3-sppadd-9-cb82		add	[25, 15]	[32, 64]	[234, 207]
YOLOv3-add-9-cb82					
YOLOv3-3-cb82	Train/Val Split based on Class-Balanced (8:2)	concatenation			
YOLOv3-spp-3-cb82					
YOLOv3-sppadd-3-cb82		add	[21, 18]	[39, 57]	[234, 207]
Balanced-YOLOv3 (YOLOv3-add-3-cb82)					

* YOLOv3 and YOLOv3-spp in the algorithm name represent two baseline algorithms, respectively. The appearance of add in the name means that the additional feature fusion strategy is adopted, and the absence of add means that the concatenation feature fusion strategy is adopted. 9 and 3 represent dense anchors and efficient anchors, respectively. The random at the end of the name represents the Train/Val generated by randomly splitting according to the ratio of photos number, and cb82 represents the Train/Val using class-balanced.

4.4.2. Objective Analysis of Experimental Results

For the PCB assembly scene containing 21 classes of detection objects, we use AP (Average Precision) to represent the probability of correct detection of a single class in the algorithm and mAP (mean Average Precision) to represent the average detection accuracy of all classes in the algorithm. The higher AP and mAP of an algorithm, the better the recognition performance of the algorithm to the object. For the object detection problem, the detection algorithm will output a prediction box to identify the position of the detected object. The IoU between the prediction box and the ground truth of the object can indicate the accuracy of the detection. Usually, a fixed IoU threshold of 0.5 is used to calculate the AP value. At the same time, to quantify the accuracy of the prediction box positioning, the AP average value is calculated for multiple IoU thresholds. Specifically, 10 IoU thresholds

between 0.5 and 0.95 (0.5, 0.55, 0.6, . . . 0.9, 0.95) to calculate the mean value of mAP. In this paper, mAP@.5 and mAP@.5:.95 represent the recognition accuracy and positioning accuracy. Based on the randomly split Train/Val ratio on the photo number (8:2), Table 4 counts the AP, mAP@.5, and mAP@.5:.95 tested for 21 classes of the test set after eight groups of algorithms are trained.

Table 4. AP, mAP@.5, and mAP@.5:.95 of eight algorithms based on Train/Val random.

Class	AP of Eight Algorithm							
	YOLOv3-9-random	YOLOv3-spp-9-random	YOLOv3-sppadd-9-random	YOLOv3-add-9-random	YOLOv3-3-random	YOLOv3-spp-3-random	YOLOv3-sppadd-3-random	YOLOv3-add-3-random
30CTQ	99.56%	99.55%	99.55%	99.56%	99.56%	99.54%	99.55%	99.55%
AUIRFR	92.52%	98.29%	93.14%	96.37%	91.87%	95.69%	95.68%	96.98%
BUK7608	99.54%	99.55%	99.55%	99.55%	99.55%	99.55%	99.55%	99.55%
Cap100uF	86.37%	90.64%	90.28%	88.33%	91.46%	89.96%	91.92%	90.43%
Cap220uF	93.78%	93.75%	91.63%	95.24%	94.85%	95.12%	93.63%	92.81%
Cap22uF	89.46%	91.55%	89.62%	89.12%	85.94%	89.67%	90.34%	91.05%
Cap470uF	92.91%	94.43%	94.85%	93.22%	95.21%	92.74%	94.64%	93.01%
GK835	85.97%	99.52%	99.52%	88.11%	99.52%	99.52%	99.52%	99.53%
Inductance	89.75%	90.63%	91.79%	89.88%	92.65%	91.75%	92.46%	93.86%
Inserted100uF	76.85%	77.23%	80.69%	72.84%	80.97%	78.13%	78.39%	79.38%
Inserted220uF	99.55%	99.55%	99.55%	99.44%	99.54%	99.54%	99.44%	99.54%
Inserted22uF	91.87%	90.95%	91.01%	86.13%	87.73%	80.45%	93.91%	90.01%
Inserted470uF	82.32%	89.63%	92.04%	92.39%	99.53%	89.27%	99.54%	99.53%
InsertedInd	39.20%	28.63%	85.85%	83.51%	61.55%	87.20%	85.04%	94.80%
PCB	99.52%	99.52%	99.52%	99.52%	99.53%	99.53%	99.53%	99.52%
Pin100uF	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.51%
Pin220uF	93.69%	92.92%	93.31%	93.38%	93.36%	93.03%	93.47%	93.06%
Pin22uF	99.51%	99.51%	85.51%	99.51%	99.51%	99.51%	99.51%	99.51%
Pin470uF	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%
PinInd	92.39%	97.76%	72.72%	73.85%	88.89%	97.76%	77.38%	72.52%
SSG8	81.51%	81.51%	81.51%	81.51%	82.28%	82.70%	83.16%	82.89%
mAP@.5	89.78%	91.15%	91.94%	91.45%	92.50%	93.32%	93.60%	93.65%
mAP@.5:.95	49.54%	49.87%	50.68%	51.17%	50.41%	51.48%	50.39%	52.47%

* The bold indicates that this algorithm's result is better than or equal to other algorithms. The YOLOv3-add-3-random algorithm achieved the best results in 9 out of 23 detection indicators. Among the eight algorithms based on Train/Val random, the YOLOv3-add-3-random was the best for recognition and position performance.

As can be seen from Table 4, under the preset box of 9 anchors, the algorithm using the addition feature fusion strategy is superior to the concatenation feature fusion strategy of the corresponding algorithm in both recognition accuracy and positioning accuracy. At the same time, the object detection effect of the preset box of 3 anchors is generally better than that of the algorithm corresponding to 9 anchors. YOLOv3-add-3-random with addition feature fusion and three anchors achieves the best detection accuracy.

Based on class balance (8:2) Train/Val, Table 5 counts the AP, mAP@.5, and mAP@.5:.95 under eight groups of algorithms for 21 classes in the test set. Comparing Tables 4 and 5, we find that the detection performance based on Train/Val cb82 is always better than Train/Val random when using the same feature fusion or anchor for the same test set. The Balanced-YOLOv3 proposed in this paper achieves the best object recognition and position accuracy.

Table 5. AP, mAP@.5, and mAP@.5:.95 of eight algorithms based on Train/Val cb82.

Class	AP of Eight Algorithm							Balanced-YOLOv3 (YOLOv3-add-3-cb82)
	YOLOv3-9-cb82	YOLOv3-spp-9-cb82	YOLOv3-sppadd-9-cb82	YOLOv3-add-9-cb82	YOLOv3-3-cb82	YOLOv3-spp-3-cb82	YOLOv3-sppadd-3-cb82	
30CTQ	99.56%	99.55%	99.55%	99.55%	99.56%	99.56%	99.55%	99.55%
AUIRFR	95.68%	96.37%	93.14%	97.60%	95.68%	95.06%	93.14%	96.38%
BUK7608	99.55%	99.55%	99.54%	99.55%	99.55%	99.54%	99.54%	99.54%
Cap100uF	91.35%	90.36%	84.54%	87.67%	90.98%	91.20%	87.03%	91.08%
Cap220uF	92.64%	94.45%	90.53%	92.50%	92.75%	93.58%	90.61%	93.83%
Cap22uF	92.18%	90.33%	90.49%	90.20%	89.26%	88.54%	89.40%	88.71%
Cap470uF	93.47%	95.59%	93.88%	93.47%	94.91%	92.21%	94.57%	93.35%
GK835	99.52%	86.59%	99.52%	99.52%	99.52%	99.52%	99.52%	99.53%
Inductance	93.71%	91.82%	91.44%	91.06%	91.98%	93.31%	92.22%	89.32%
Inserted100uF	59.76%	77.33%	82.97%	74.37%	80.88%	76.60%	78.51%	77.88%
Inserted220uF	99.55%	99.54%	99.55%	99.54%	99.54%	99.55%	99.55%	99.54%
Inserted22uF	82.44%	91.01%	87.67%	91.60%	85.65%	86.26%	92.35%	92.03%
Inserted470uF	92.39%	99.53%	99.53%	93.09%	99.54%	99.53%	93.10%	99.53%
InsertedInd	92.24%	67.13%	90.90%	97.23%	88.31%	89.16%	90.90%	97.23%
PCB	99.53%	99.52%	99.52%	99.52%	99.52%	99.53%	99.52%	99.53%
Pin100uF	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%
Pin220uF	92.57%	93.03%	92.98%	93.07%	93.75%	93.65%	93.03%	92.92%
Pin22uF	99.51%	99.51%	99.51%	99.51%	99.51%	99.51%	99.51%	99.51%
Pin470uF	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%	99.52%
PinInd	90.64%	92.39%	96.40%	88.26%	73.85%	88.89%	90.26%	88.89%
SSG8	82.47%	81.51%	81.51%	81.51%	81.51%	83.29%	81.51%	81.98%
mAP@0.5	92.75%	92.58%	93.91%	93.71%	93.11%	93.69%	93.47%	94.25%
mAP@.5:.95	51.17%	51.82%	51.59%	52.28%	52.63%	53.76%	51.14%	54.20%

* The bold indicates that this algorithm's result is better than or equal to other algorithms. Among the eight algorithms based on Train/Val cb82, the Balanced-YOLOv3 was the best for recognition and position performance.

4.4.3. Analysis of the Experimental Process

To further compare the advantages and disadvantages of the above 16 algorithms, we draw six curves for comparative analysis, as shown in Figure 11:

Figure 11a,b shows the variation of the recognition and localization indicators of the validation set with the training epoch of the 16 algorithms during model training. The mAP is used as the real bounding box and the predicted box to compare and return the score value. The higher the score, the higher the model's object recognition accuracy. The mAP@.5:.95 is taken as the average value of mAP under 10 IoU thresholds. Also, the higher the score, the higher the accuracy of the model's localization of the object. Figure 11a,b shows that Balanced-YOLOv3 can obtain the best performance of recognition and localization for different validation sets under the split of two Train/Val.

Figure 11c,d shows the variation of the loss function values of the training set and the validation set with the training epoch, respectively, during model training for 16 algorithms. The faster the loss value decreases, the more stable the convergence value is. It shows that the lower the complexity of the algorithm, the shorter the training time of the algorithm. In Figure 11c, the training loss value is naturally divided into three regions during the training process. The algorithm with three anchors declines the fastest, with nine anchors having a higher final loss value. In Figure 11d, the validation loss value is naturally divided into four regions during the validation process. The loss value of the validation set based on the class-balanced decreases rapidly, and the loss value of the validation set randomly decreases slowly. Under the same validation set, the algorithm with three anchors descends faster than with nine anchors. Regardless of training set loss or validation set loss, Balanced-YOLOv3 is the algorithm with the fastest drop and the lowest numerical value.

Figure 11e is the precision-recall curve; the precision is the probability that the predicted bounding box matches the actual ground-truth box, also known as the positive predicted value. The value ranges from 0 to 1, and high precision means most detected objects match the real objects. The recall represents the probability of correctly detecting the

true object. Similarly, the recall value ranges from 0 to 1, and a high recall means that many real objects were detected. The higher the precision-recall curve, the more towards the upper right corner, which means the algorithm has high precision and high recall and can detect most real objects correctly. The Balanced-YOLOv3 algorithm has the highest position in Figure 11e and is closest to the upper right corner, and the detection effect is good.

The F1 metric in Figure 11f considers both the precision and recall of the classification model and can measure the balance between the recall and the precision. The core idea of F1 is to improve the precision to recall as much as possible and hope that the difference between the two is as small as possible. When the value of F1 is high, it means that both precision and recall are high. A low F1 score means a significant imbalance between precision and recall. It can be seen from Figure 11f that Balanced-YOLOv3 has the highest F1 value, indicating that the algorithm can take into account both high precision and high recall.

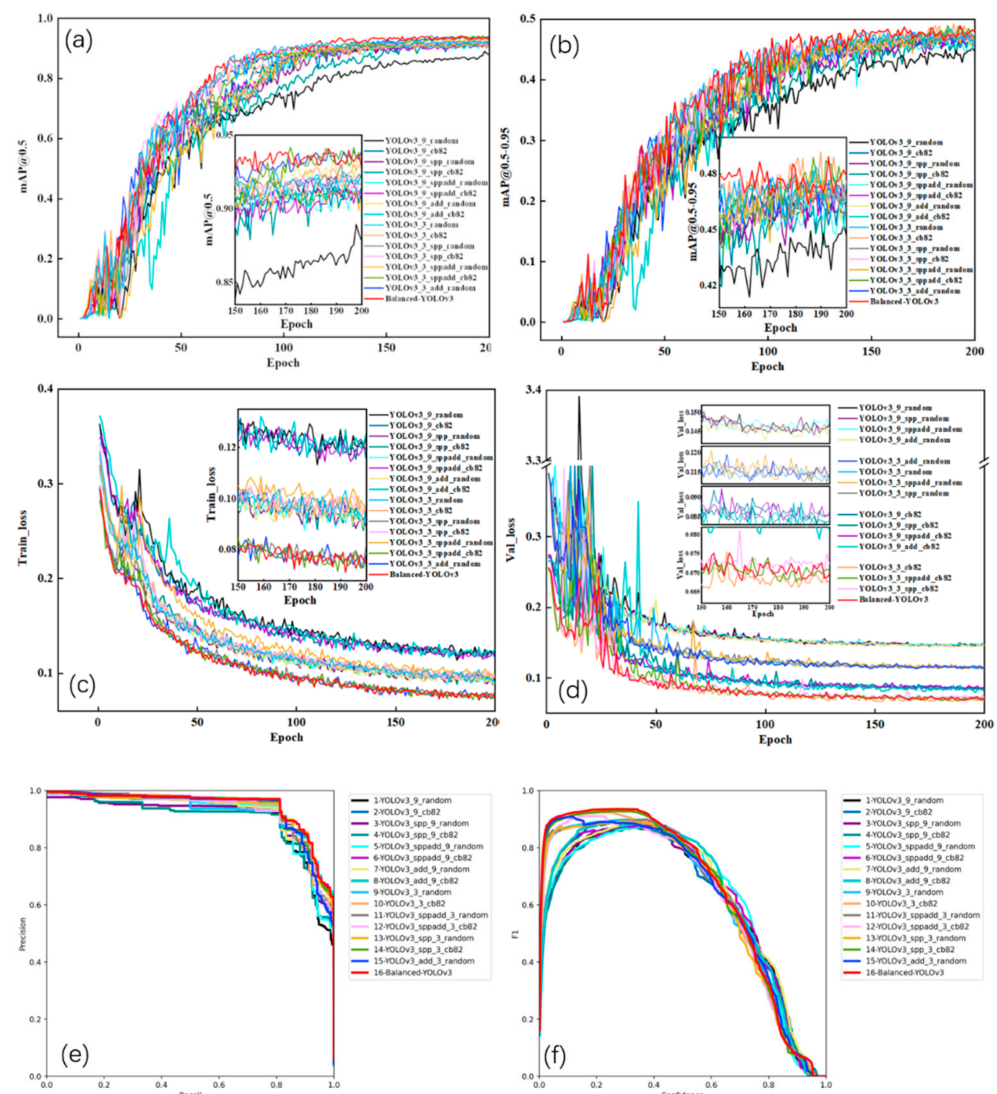


Figure 11. Sixteen sets of algorithm experiment process curve diagrams. (a) mAP@0.5-Epoch change trend curve. (b) mAP@0.5:0.95-Epoch change trend curve. (c) Train_loss-Epoch change trend curve. (d) Val_loss-Epoch change trend curve. (e) Precision-Recall curve. (f) F1 curve.

4.4.4. Trade-Off Analysis of Algorithmic Reasoning Time and Parameter Amount

For the object detection algorithm, we hope that it has a high detection accuracy rate and a fast detection speed, and a small number of parameters. We visualize the scatter plots of inference time and parameters for the above 16 algorithms, as shown in Figure 12.

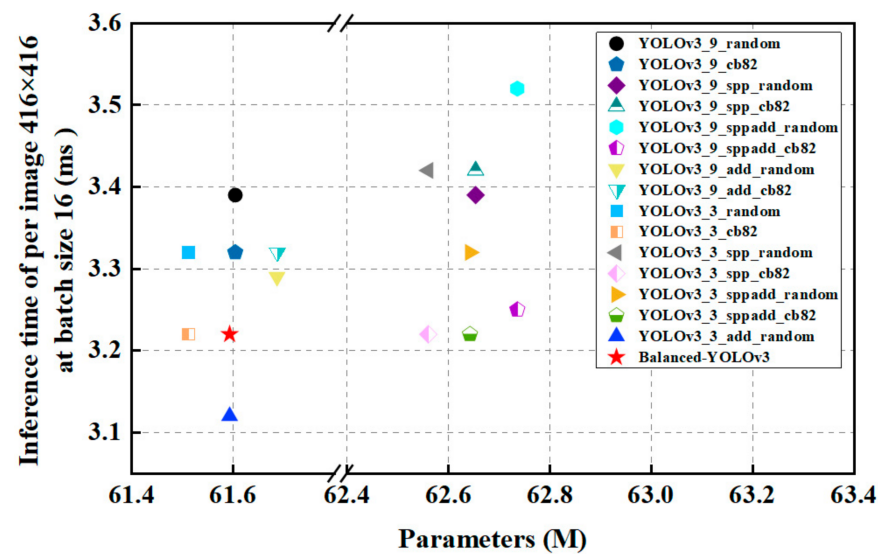


Figure 12. Parameter-Inference Time diagram.

The horizontal axis of the scatter plot is the algorithm parameter, and the vertical axis is the inference time of a single photo when the photo in the test set is resized to 416×416 , and the batch size is 16. A lightweight and efficient object detection algorithm should have low parameters and low inference time and should be located in the bottom left corner of the scatter plot. Figure 12 shows that YOLOv3-3-random and YOLOv3-3-cb82 have the fewest parameters; they are the lightest. YOLOv3-add-3-random has the shortest inference time for test set pictures and the most efficient detection. Balanced-YOLOv3 ranks third in the lightweight and efficiency metrics.

4.5. Ablation Experiment

Aiming at Balanced-YOLOv3 proposed in this paper to solve the problem of imbalanced object detection in PCB assembly scenes, there are three modules of class-balanced Train/Val, addition feature fusion, and efficient anchors. We pass the ablation experiments in Table 6 to investigate the contribution of these modules to the Balanced-YOLOv3. The symbol \times means not to use this module, and the symbol \checkmark means to use this module.

Table 6. Effectiveness of each module in Balanced-YOLOv3.

Algorithm	Class-Balanced Train/Val	Add	Efficient Anchor	mAP@.5	mAP@.5:.95
YOLOv3-9-random	\times	\times	\times	89.78%	49.54%
YOLOv3-9-cb82	\checkmark	\times	\times	92.75%+2.97%	51.17%+1.63%
YOLOv3-add-9-random	\times	\checkmark	\times	91.45%+1.67%	51.17%+1.63%
YOLOv3-3-random	\times	\times	\checkmark	92.50%+2.72%	50.41%+0.87%
YOLOv3-add-3-random	\times	\checkmark	\checkmark	93.65%+3.87%	52.47%+2.93%
YOLOv3-add-9-cb82	\checkmark	\checkmark	\times	93.71%+3.93%	52.28%+2.74%
YOLOv3-3-cb82	\checkmark	\times	\checkmark	93.11%+3.33%	52.63%+3.09%
Balanced-YOLOv3	\checkmark	\checkmark	\checkmark	94.25%+4.47%	54.20%+4.66%

Here we use mAP@.5 and mAP@.5:.95 as the evaluation metric to measure the degree of contribution. As shown in Table 6, for the effectiveness of a single module, the class-balanced Train/Val has the most significant contribution to mAP@.5, followed by efficient anchors, the smallest contribution is the added feature fusion strategy. Class-balanced Train/Val and addition feature fusion strategy contribute equally to mAP@.5:.95, with a relatively small contribution from efficient anchors. For the combined use of pairwise modules, we find that the combined use of class-balanced and addition feature fusion contributes the most to mAP@.5. The combined use of addition feature fusion and efficient anchors contribute the most middle. The combined use of class-balanced and efficient

4.7. Performance Comparison of Balanced-YOLOv3 and Other Object Detection Algorithms

To further compare the performance of the algorithms, we compare and analyze Balanced-YOLOv3 with other current state-of-the-art anchor-based object detection methods, including Faster R-CNN, SSD [51], YOLOv4 [52], and YOLOv5, for the Train/Val with class-balanced. The indicators of comparison are detection accuracy and computational complexity. The detection accuracy is represented by mAP, and the computational complexity is represented by the algorithm's parameter and GFLOPs (1 billion floating-point operations per second). The number of algorithm parameters defines the storage space required to store the algorithm model, and GFLOPs represents the algorithm's computational power. The lower the complexity of the CNN, the less the number of parameters required and the lower the computing power.

Table 7 summarizes the numerical values of Balanced-YOLOv3 and other current object detection algorithms on mAP, parameter, and GFLOPs. It can be seen from the table that the mAP values of the YOLO series of algorithms are higher than those of the traditional Faster-RCNN and SSD. At the same time, the mAP of the YOLO algorithm based on three anchors is higher than that of the YOLO algorithm based on nine anchors. Moreover, the complexity of the YOLO algorithm based on three anchors is lower than the YOLO algorithm based on nine anchors in the same series. At the same time, we also tested that after adding the added feature fusion to YOLOv5 using three anchors, their computational complexity is reduced, the mAP of YOLOv5-L-add(3_anchor) and YOLOv5-m-add(3_anchor) increased. Compared with other algorithms, Balanced-YOLOv3 has the highest mAP, and the algorithm complexity is close to the lowest. Table 7 means the Balanced-YOLOv3 is a high-precision, low-complexity object detection algorithm suitable for the PCB assembly scene.

Table 7. Statistics of accuracy and complexity of eleven algorithms.

Model	mAP	Params	GFLOPs
Faster R-CNN (Resnet50)	67.71%	43.44	742.47
SSD(VGG16)	75.18%	28.52	91.55
YOLOv4(9_anchor)	87.83%	64.05	90.74
YOLOv4(3_anchor)	88.80%	63.17	90.49
YOLOv5-s(9_anchor)	78.04%	7.07	16.0
YOLOv5-s(3_anchor)	91.02%	7.02	15.8
YOLOv5-s-add(3_anchor)	88.27%	6.84	15.4
YOLOv5-m(9_anchor)	87.72%	20.93	48.2
YOLOv5-m(3_anchor)	93.18%	20.86	48.0
YOLOv5-m-add(3_anchor)	93.14%	20.46	47.0
YOLOv5-l(9_anchor)	91.52%	46.22	108.1
YOLOv5-l(3_anchor)	93.44%	46.12	107.8
YOLOv5-l-add(3_anchor)	93.54%	45.40	106.2
Balanced-YOLOv3	94.25%	61.59	32.69

* The bold indicates that this algorithm's result is better than or equal to other algorithms.

4.8. Discuss

We have proved that the Train /Val with the balanced-class split for PCB assembly scene object detection functions. However, it cannot directly solve the class imbalance phenomenon in the dataset, and it can guarantee that each class object is in the training set and validation set. The number ratio is roughly equal to avoid missing classes in the Train/Val based on the number ratio of photos. Missing classes or imbalanced class ratios will directly lead to the deviation of CNN parameter learning. Our proposed addition feature fusion strategy enhances the multi-scale imbalanced features of the object through adding. Compared with the traditional concatenation feature fusion strategy, it strengthens the location information of the CNN deep network and the semantic information of the shallow network, and the number of channels is reduced, reducing the parameters. The

efficient anchor we designed considers the object's size and combines the multi-scale spatial distribution characteristics of the object. Two clusters generate it. Compared with the traditional nine dense anchors, the efficient anchor reduces redundant anchors and alleviates the imbalance of positive/negative samples. Based on the dataset split by two Train/Val, we compared the sixteen algorithms' objective statistics and training process trend graphs and compared mAP and algorithm complexity with other anchor-based object detection algorithms. We found that Balanced-YOLOv3 is an excellent performance in the PCB assembly scene object detection.

It should be noted that class imbalance, multi-scale feature imbalance, and positive/negative sample imbalance exist not only in PCB assembly scene object detection tasks but also in other object detection tasks, such as pedestrian detection, face detection, object detection in optical remote sensing images, and vehicle detection, etc. Therefore, our proposed Balanced-YOLOv3, combined with the data characteristics of other detection objects, can also bring some inspiration to improve the detection effect for other object detection tasks.

Although Balanced-YOLOv3 has the advantages of high detection accuracy and low computational complexity, there are still some potential limitations and challenges to improving its effectiveness further. First of all, for the problem of object class imbalance, the class-balanced Train/Val split method proposed in this paper starts from the perspective of fair learning of different classes of objects in training and verification. Still, it does not consider the size of the object. If we use the size as the weight of the Train/Val split, CNN can better learn the features of small-sized objects that are difficult to detect. However, quantifying the contribution of different size objects to the Train/Val is a problem. Secondly, for the imbalance problem of multi-scale features, the additional feature fusion strategy proposed in this paper starts from the perspective of strengthening the deep object location information and shallow object semantic information, completely abandoning the concatenation feature fusion method and the combined use of concatenation and addition feature fusion methods is not considered. The embedding position and order of dual feature fusion in CNN is a complex problem to improve object detection. Finally, the proposal of efficient anchors reduces the redundant information brought by dense anchors and alleviates the imbalance of positive/negative samples but does not consider the affiliation between the object and the object in the PCB assembly scene. It is difficult to strengthen positive samples and weaken negative samples as a knowledge guiding condition. Fortunately, based on recent experience, the current state of research on uniform distribution alignment strategies [53], cascaded hourglass feature fusion [54], and knowledge graphs [55] can all bring inspiration to address the above problems. Therefore, future research related to the imbalance problem in object detection will deepen in establishing a unified method to adapt to different objects, the in-depth study of specific imbalance problems, and the improvement of detection results and detection speed.

5. Conclusions

In this paper, the object detection in the PCB assembly scene includes large-sized PCBs, medium-sized plug-in electronic components, and small-sized through-holes, comprising 29 classes of objects. The scenarios faced include before, during, and after the assembly of through-hole electronic components. Aiming at the problem of imbalanced object classes in the dataset, the Train /Val balanced split method proposed in this paper, without adding any data to the dataset, splits all classes of objects into the training set and the validation set in a balanced manner. The CNN can learn the features of all classes during the training process and find the model parameter combination with the best performance during the verification process. This Train/Val balanced split method establishes a new idea for solving the problem of class imbalance. Aiming at the imbalance of object feature scales extracted by the backbone, this paper proposes a fusion method that uses feature addition instead of feature concatenation, which strengthens the features of object location information and semantic information, and reduces the amount of data in the model.

The problem of disappearing or weakening object features in the propagation process due to feature scale imbalance is solved. Because of the imbalance of positive/negative samples, this paper proposes the concept of the efficient anchor. The number of efficient anchors for each detection head is determined starting from the detection dataset, taking into account the object size and the spatial position of the three output scales. It avoids the data redundancy problem caused by dense anchors and fundamentally solves the imbalance problem of positive/negative samples. The Balanced-YOLOv3 object detector that integrates these three contributions can well handle the imbalance problem in object detection in PCB assembly scenes. Experiments show that compared with other anchor-based object detection benchmarks, Balanced-YOLOv3 has the highest detection accuracy and low computational complexity, which provides ideas for the future research direction of intelligent perception problems in electronic manufacturing.

Author Contributions: Conceptualization, J.L.; Data curation, J.L.; Formal analysis, J.L. and Y.C.; Investigation, Y.C.; Methodology, J.L.; Resources, J.L. and W.L.; Software, W.L. and Y.C.; Supervision, J.G.; Writing—original draft, J.L.; Writing—review & editing, J.L., Y.C., W.L., and J.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 51875266).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare they have no conflict of interest.

References

1. Yang, T.; Yi, X.L.; Lu, S.W.; Johansson, K.H.; Chai, T.Y. Intelligent Manufacturing for the Process Industry Driven by Industrial Artificial Intelligence. *Engineering* **2021**, *7*, 1224–1230. [\[CrossRef\]](#)
2. Yang, T.; Ding, J.L.; Vamvoudakis, K.G.; Qin, S.J. Guest Editorial: Industrial Artificial Intelligence for Smart Manufacturing. *IEEE Trans. Ind. Inform.* **2021**, *17*, 8319–8323. [\[CrossRef\]](#)
3. Senoner, J.; Netland, T.; Feuerriegel, S. Using Explainable Artificial Intelligence to Improve Process Quality: Evidence from Semiconductor Manufacturing. *Manag. Sci.* **2021**, 1–20. [\[CrossRef\]](#)
4. Mozaffar, M.; Liao, S.H.; Xie, X.Y.; Saha, S.; Park, C.; Cao, J.; Liu, W.K.; Gan, Z.T. Mechanistic artificial intelligence (mechanistic-AI) for modeling, design, and control of advanced manufacturing processes: Current state and perspectives. *J. Mater. Processing Technol.* **2022**, *302*, 117485. [\[CrossRef\]](#)
5. Kim, S.W.; Kong, J.H.; Lee, S.W.; Lee, S. Recent Advances of Artificial Intelligence in Manufacturing Industrial Sectors: A Review. *Int. J. Precis. Eng. Manuf.* **2022**, *23*, 111–129. [\[CrossRef\]](#)
6. Li, J.; Li, W.; Chen, Y.; Gu, J. Research on Object Detection of PCB Assembly Scene Based on Effective Receptive Field Anchor Allocation. *Comput. Intell. Neurosci.* **2022**, *2022*, 32. [\[CrossRef\]](#)
7. Sahu, C.K.; Young, C.; Rai, R. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: A review. *Int. J. Prod. Res.* **2021**, *59*, 4903–4959. [\[CrossRef\]](#)
8. Oksuz, K.; Cam, B.C.; Kalkan, S.; Akbas, E. Imbalance Problems in Object Detection: A Review. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 3388–3415. [\[CrossRef\]](#)
9. Ren, J.; Yu, C.; Sheng, S.; Ma, X.; Zhao, H.; Yi, S.; Li, H. Balanced Meta-Softmax for Long-Tailed Visual Recognition. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada, 6–12 December 2020.
10. Wang, T.; Li, Y.; Kang, B.; Li, J.; Liew, J.; Tang, S.; Hoi, S.; Feng, J. The devil is in classification: A simple framework for long-tail instance segmentation. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 728–744.
11. Wang, Y.; Gan, W.; Yang, J.; Wu, W.; Yan, J. Dynamic curriculum learning for imbalanced data classification. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 5017–5026.
12. Kang, B.; Xie, S.; Rohrbach, M.; Yan, Z.; Gordo, A.; Feng, J.; Kalantidis, Y. Decoupling Representation and Classifier for Long-Tailed Recognition. In Proceedings of the International Conference on Learning Representations, Ababa, Ethiopia, 26 April–1 May 2020.
13. Zhang, Z.; Pfister, T. Learning fast sample re-weighting without reward data. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, Canada, 11–17 October 2021; pp. 725–734.
14. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.

15. Zhao, Q.; Sheng, T.; Wang, Y.; Tang, Z.; Chen, Y.; Cai, L.; Ling, H. M2Det: A single-shot object detector based on Multi-Level Feature Pyramid Network. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; p. 1136.
16. Xu, A.; Yao, A.; Li, A.; Liang, A.; Zhang, A. Auto-FPN: Automatic Network Architecture Adaptation for Object Detection Beyond Classification. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 6648–6657.
17. Kong, T.; Sun, F.; Huang, W.-b.; Liu, H. Deep Feature Pyramid Reconfiguration for Object Detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
18. Kong, T.; Sun, F.; Yao, A.; Liu, H.; Lu, M.; Chen, Y. RON: Reverse Connection with Objectness Prior Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 1–26 July 2017; pp. 5244–5252.
19. Kim, S.-W.; Kook, H.-K.; Sun, J.-Y.; Kang, M.-C.; Ko, S.-J. Parallel Feature Pyramid Network for Object Detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; Springer International Publishing: Cham, Switzerland; Volume 2018, pp. 239–256.
20. Li, H.; Liu, Y.; Ouyang, W.; Wang, X. Zoom Out-and-In Network with Map Attention Decision for Region Proposal and Object Detection. *Int. J. Comput. Vis.* **2019**, *127*, 225–238. [\[CrossRef\]](#)
21. Ge, Z.; Jie, Z.Q.; Huang, X.; Li, C.Z.; Yoshie, O. Delving deep into the imbalance of positive proposals in two-stage object detection. *Neurocomputing* **2021**, *425*, 107–116. [\[CrossRef\]](#)
22. Han, Z.S.; Wang, C.P.; Fu, Q. (MR)-R-2-Net: Deep network for arbitrary oriented vehicle detection in MiniSAR images. *Eng. Comput.* **2021**, *38*, 2969–2995. [\[CrossRef\]](#)
23. Hou, X.Y.; Zhang, K.L.; Xu, J.H.; Huang, W.; Yu, X.M.; Xu, H.Y. Object Detection in Drone Imagery via Sample Balance Strategies and Local Feature Enhancement. *Appl. Sci.* **2021**, *11*, 3547. [\[CrossRef\]](#)
24. Li, H.; Chen, L.; Han, H.; Chi, Y.; Zhou, S.K. Conditional Training with Bounding Map for Universal Lesion Detection. In Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), Electr Network, Strasbourg, France, 27 September–1 October 2021; pp. 141–152.
25. Li, N.; Lyu, X.; Xu, S.K.; Wang, Y.R.; Wang, Y.S.; Gu, Y.W. Incorporate Online Hard Example Mining and Multi-Part Combination Into Automatic Safety Helmet Wearing Detection. *IEEE Access* **2021**, *9*, 139536–139543. [\[CrossRef\]](#)
26. Li, Z.H.; Zhuang, X.P.; Wang, H.B.; Nie, Y.; Tang, J.Z. Local Attention Sequence Model for Video Object Detection. *Appl. Sci.* **2021**, *11*, 4561. [\[CrossRef\]](#)
27. Li, Z.Y.; Wang, H.J.; Zhong, H.F.; Dai, Y.T. Self-attention module and FPN-based remote sensing image target detection. *Arab. J. Geosci.* **2021**, *14*, 18. [\[CrossRef\]](#)
28. Xu, T.; Sun, X.; Diao, W.H.; Zhao, L.J.; Fu, K.; Wang, H.Q. ASSD: Feature Aligned Single-Shot Detection for Multiscale Objects in Aerial Imagery. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 17. [\[CrossRef\]](#)
29. Li, Q.Z.; Zhang, Y.J.; Sun, S.Y.; Zhao, X.G.; Li, K.; Tan, M. Rethinking semantic-visual alignment in zero-shot object detection via a softplus margin focal loss. *Neurocomputing* **2021**, *449*, 117–135. [\[CrossRef\]](#)
30. Li, Y.S.; Liu, C.L.; Shen, Y.; Cao, J.; Yu, S.C.; Du, Y.C. RoadID: A Dedicated Deep Convolutional Neural Network for Multipavement Distress Detection. *J. Transp. Eng. Part B-Pavements* **2021**, *147*, 12. [\[CrossRef\]](#)
31. Li, Y.T.; Wu, Z.H.; Li, L.; Yang, D.N.; Pang, H.F. Improved YOLOv3 model for vehicle detection in high-resolution remote sensing images. *J. Appl. Remote Sens.* **2021**, *15*, 15. [\[CrossRef\]](#)
32. Lu, X.C.; Ji, J.; Xing, Z.Q.; Miao, Q.G. Attention and Feature Fusion SSD for Remote Sensing Object Detection. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 9. [\[CrossRef\]](#)
33. Zheng, D.C.; Zhang, Y.Z.; Xiao, Z.J. Deep Learning-Driven Gaussian Modeling and Improved Motion Detection Algorithm of the Three-Frame Difference Method. *Mob. Inf. Syst.* **2021**, *2021*, 7. [\[CrossRef\]](#)
34. Li, X.H.; He, M.Z.; Li, H.F.; Shen, H.F. A Combined Loss-Based Multiscale Fully Convolutional Network for High-Resolution Remote Sensing Image Change Detection. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 5. [\[CrossRef\]](#)
35. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *1*, 91–99. [\[CrossRef\]](#) [\[PubMed\]](#)
36. Pang, J.; Chen, K.; Shi, J.; Feng, H.; Ouyang, W.; Lin, D. Libra r-cnn: Towards balanced learning for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 821–830.
37. Li, J.; Gu, J.N.; Huang, Z.D.; Wen, J. Application Research of Improved YOLO V3 Algorithm in PCB Electronic Component Detection. *Appl. Sci.* **2019**, *9*, 3750. [\[CrossRef\]](#)
38. Liu, C.; Liu, S.Q. Tiny Electronic Component Detection Based on Deep Learning. In Proceedings of the 5th International Conference on Green Power, Materials and Manufacturing Technology and Applications (GPMMTA), Taiyuan, China, 21–22 September 2019.
39. Zhao, Y.T.; Zheng, B.; Li, H.C. FRCNN-Based DL Model for Multiview Object Recognition and Pose Estimation. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 9487–9494.
40. Tsai, D.M.; Chou, Y.H. Fast and Precise Positioning in PCBs Using Deep Neural Network Regression. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 4692–4701. [\[CrossRef\]](#)

41. Lin, Y.L.; Chiang, Y.M.; Hsu, H.C. Capacitor detection in PCB using YOLO algorithm. In Proceedings of the International Conference on System Science and Engineering (ICSSE), Taipei, Taiwan, 28–30 June 2018.
42. Kuo, C.W.; Ashmore, J.D.; Huggins, D.; Kira, Z. Data-Efficient Graph Embedding Learning for PCB Component Detection. In Proceedings of the 19th IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa Village, HI, USA, 7–11 January 2019; pp. 551–560.
43. Lu, Y.Q.; Yang, B.; Gao, Y.C.; Xu, Z.M. An automatic sorting system for electronic components detached from waste printed circuit boards. *Waste Manag.* **2022**, *137*, 1–8. [\[CrossRef\]](#)
44. Mukhopadhyay, A.; Mukherjee, I.; Biswas, P. Comparing shape descriptor methods for different color space and lighting conditions. *AI EDAM-Artif. Intell. Eng. Des. Anal. Manuf.* **2019**, *33*, 389–398. [\[CrossRef\]](#)
45. Liu, X.; Hu, J.S.; Wang, H.X.; Zhang, Z.G.; Lu, X.; Sheng, C.Y.; Song, S.B.; Nie, J. Gaussian-IoU loss: Better learning for bounding box regression on PCB component detection. *Expert Syst. Appl.* **2022**, *190*, 11. [\[CrossRef\]](#)
46. Baranwal, A.; Meyer, M.; Nguyen, T.; Pillai, S.; Nakayamada, N.; Wahlsten, M.; Fujimura, A.; Niewczas, M.; Pomerantsev, M. Five Deep Learning Recipes for the Mask-making Industry. In Proceedings of the Photomask Technology Conference, Monterey, CA, USA, 16–18 September 2018.
47. Li, J.; Li, W.Y.; Chen, Y.Q.; Gu, J.A. A PCB Electronic Components Detection Network Design Based on Effective Receptive Field Size and Anchor Size Matching. *Comput. Intell. Neurosci.* **2021**, *2021*, 6682710. [\[CrossRef\]](#)
48. Shen, J.Q.; Liu, N.Z.; Sun, H. Defect detection of printed circuit board based on lightweight deep convolution network. *IET Image Process.* **2020**, *14*, 3932–3940. [\[CrossRef\]](#)
49. Guo, C.; Lv, X.L.; Zhang, Y.; Zhang, M.L. Improved YOLOv4-tiny network for real-time electronic component detection. *Sci. Rep.* **2021**, *11*, 13. [\[CrossRef\]](#) [\[PubMed\]](#)
50. Shuai, Y.; Yang, C.; Chen, J.; Yuan, C.; Song, T.L. Secondary Screening Detection Optimization Method for Electronic Components Based on Artificial Intelligence. In Proceedings of the 10th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 18–20 October 2019; pp. 353–357.
51. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
52. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
53. Zhang, S.; Li, Z.; Yan, S.; He, X.; Sun, J. Distribution alignment: A unified framework for long-tail visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Montreal, QC, Canada, 11–17 October 2021; pp. 2361–2370.
54. Luo, H.; Han, G.; Wu, X.; Liu, P.; Yang, H.; Zhang, X. Cascaded hourglass feature fusing network for saliency detection. *Neurocomputing* **2021**, *428*, 206–217. [\[CrossRef\]](#)
55. Toyosaka, Y.; Okita, T.; Assoc Comp, M. Activity Knowledge Graph Recognition by Eye Gaze: Identification of Distant Object in Eye Sight for Watch Activity. In Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)/ACM International Symposium on Wearable Computers (ISWC), Electrical Network, New York, NY, USA, 21–26 September 2021; pp. 334–339.