

Article

DMA-Net: Decoupled Multi-Scale Attention for Few-Shot Object Detection

Xijun Xie ¹, Feifei Lee ^{1,*} and Qiu Chen ^{2,*}

¹ Institute of Intelligent Rehabilitation Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China; 202440420@st.usst.edu.cn

² Department of Electrical Engineering and Electronics, Graduate School of Engineering, Kogakuin University, Tokyo 163-8677, Japan

* Correspondence: feifeilee@ieeee.org (F.L.); q.chen@ieeee.org (Q.C.)

Abstract: As one of the most important fields in computer vision, object detection has undergone marked development in recent years. Generally, object detection requires many labeled samples for training, but it is not easy to collect and label samples in many specialized fields. In the case of few samples, general detectors typically exhibit overfitting and poor generalizability when recognizing unknown objects, and many FSOD methods also cannot make good use of support information or manage the potential problem of information relationships between the support branch and the query branch. To address this issue, we propose in this paper a novel framework called Decoupled Multi-scale Attention (DMA-Net), the core of which is the Decoupled Multi-scale Attention Module (DMAM), which consists of three primary parts: a multi-scale feature extractor, a multi-scale attention module, and a decoupled gradient module (DGM). DMAM performs multi-scale feature extraction and layer-to-layer information fusion, which can use support information more efficiently, and DGM can reduce the impact of potential optimization information exchange between two branches. DMA-Net can implement incremental FSOD, which is suitable for practical applications. Extensive experimental results demonstrate that DMA-Net has comparable results on generic FSOD benchmarks, particularly in the incremental FSOD setting, where it achieves a state-of-the-art performance.

Keywords: object detection; few-shot learning; incremental learning; meta learning



Citation: Xie, X.; Lee, F.; Chen, Q. DMA-Net: Decoupled Multi-Scale Attention for Few-Shot Object Detection. *Appl. Sci.* **2023**, *13*, 6933. <https://doi.org/10.3390/app13126933>

Academic Editor: Sungho Kim

Received: 6 May 2023

Revised: 27 May 2023

Accepted: 5 June 2023

Published: 8 June 2023



Copyright: © 2023 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

With the continuous progress of deep neural networks [1–19], object detection, a classic and challenging task in computer vision, has undergone many developments in recent years. However, object detection tasks all need many labeled samples for training, but in real life, sample collection and labeling in many fields are challenging, which promotes research on few-shot object detection (FSOD). The methods of using weight-shared backbones [20], calculating category prototype representations [21], judging similarity by matching inputs [22], and concatenating feature maps [23] in few-shot classification (FSC) have been widely used in previous FSOD research. However, simply transferring FSC methods cannot completely solve the FSOD task because FSOD not only needs to classify, but also localize, objects, and each image contains multiple objects of different classes. Previous studies [24–28] in recent years used an N-way K-shot training strategy based on meta learning. As shown in Figure 1, given a base class set with a large number of labeled samples and a novel class set with few samples, the query branch inputs a query image for detection and the corresponding support branch inputs N-way K-shot support images, that is, N categories and K support images per category. They used a support branch to help the network simulate the cases of a few samples, with the goal of transferring the meta-knowledge (the ability to detect objects of one category through only a few support samples of the same category) learned from the base classes to the novel classes in order to implement the recognition of objects in novel classes. How to make better use of the

information from the support branch is currently the most challenging task. Based on the widely used object detector Faster R-CNN [3], support information is used in the detection module in [25,26], but the region proposal network (RPN) is also an important part of the network. The scarcity of samples affects the foreground and background classification of the RPN, which in turn generates unsuitable proposals and affects the subsequent detection of objects. Therefore, support information was added into the query branch before RPN in [28], but was only a simple fusion of single-size features. Overall, these methods only make superficial use of support information. When sample information is scarce, it is necessary to fully utilize all aspects of support information and to consider the location and form of fusion to make full use of support information. In addition, two branch models may encounter issues of inconsistency between gradient and parameter types during parameter updates. Specifically, the RPN only classifies the foreground and background, as it is a class-agnostic module, while the RCNN must identify the category of objects, as it is a class-specific module. The support branch is also class-specific; thus, there may be a problem of unnecessary optimization information exchange between the support branch and the query branch in the models that add support information before RPN. To manage the relationship between the support branch and the query branch more effectively, and to obtain more information from the support branch while reducing the interference between the two branches, we propose the decoupled multi-scale attention (DMA-Net) method. Many existing methods require a fine-tuning stage and cannot directly recognize novel objects, while the proposed method can achieve good performance for both novel and base classes without fine-tuning, making it suitable for practical applications. To the best of our knowledge, the technique of decoupling gradients between the two branches prior to RPN has not been employed in previous FSOD models. Therefore, this study represents a pioneering effort to implement the decoupled multi-scale attention approach in FSOD models. Our source code for this method is publicly available on GitHub at: <https://github.com/xijunxie/DMA-Net>.

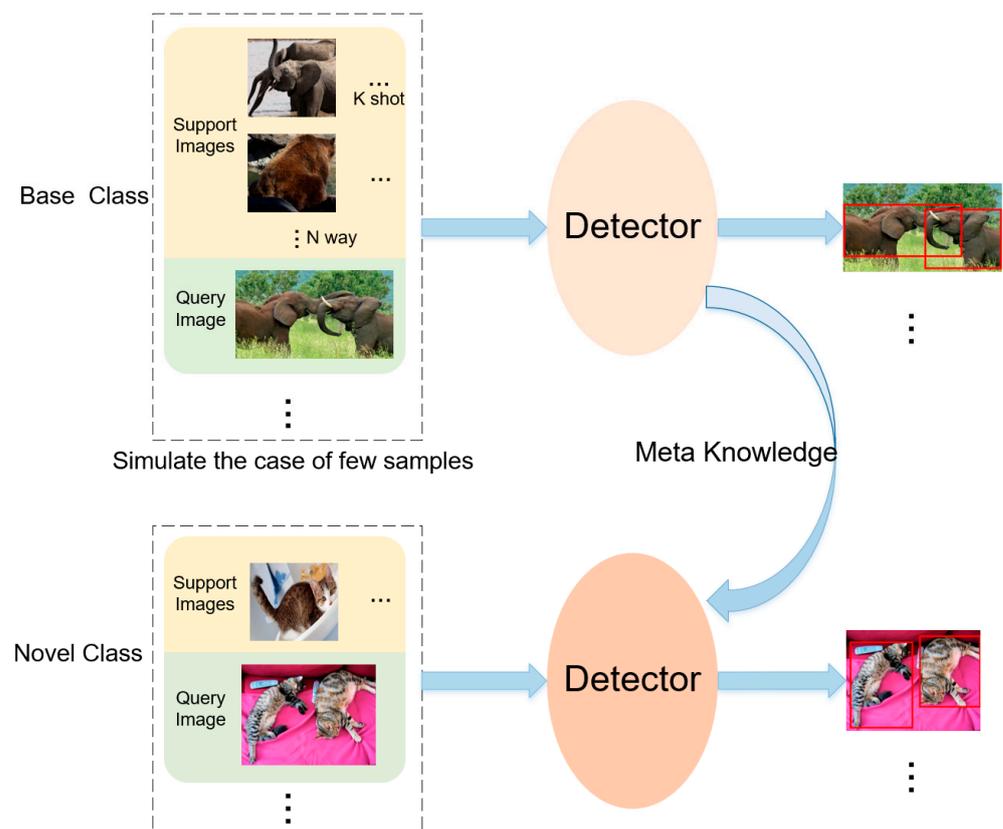


Figure 1. N-way K-shot training strategy based on meta learning.

Thus, the contributions of this study are as follows:

- (1) We propose a novel framework for few-shot object detection (FSOD) called decoupled multi-scale attention (DMA-Net) that can manage the relationship between support and query branches effectively.
- (2) We use multi-scale features for detection and overcome the problems it causes. We perform multi-scale attention from the support branch to the query branch, and more effective information and details from the support branch can be used for the query branch.
- (3) We analyze the contradiction difference between the proposed model and the original Faster R-CNN [3], and perform decoupling between the support branch and the query branch to extract support information more effectively and to reduce the mutual interference between two branches.
- (4) We evaluate the performance of the proposed network in two ways: with fine-tuning and without fine-tuning. The proposed network performs well with and without fine-tuning, but particularly without fine-tuning, achieving good performance.

The remainder of this paper is organized as follows. In Section 2, some related works on FSOD are reviewed. In Section 3, we describe the proposed method in detail. The experimental results are described in Section 4, and conclusions are given in Section 5.

2. Related Works

2.1. General Object Detection

With the rapid development of convolutional neural networks, the accuracy of object detection methods is also improving. Object detection is a classical task in computer vision that focuses on classifying and localizing objects. Object detectors based on deep learning can be divided into two categories: one-stage detectors and two-stage detectors. YOLO series [9–12] detectors and Centre Net series [4,5] detectors belong to the first category. One-stage detectors only use a few convolutional layers for classification and bounding box regression directly, which makes these detectors simpler networks with faster inference speeds. In contrast, the R-CNN series [1–3] detectors belong to the second category. The difference between two- and one-stage detectors is that the former typically have an additional module, such as RPN, to generate and select proposals based on the foreground and background in the first stage and then send the selected proposals to the detection module for additional filtering and prediction. In this case, two-stage detectors typically perform better than one-stage detectors, as they have the advantages of the additional module. However, the methods described above require many labeled examples in the visible domain and are difficult to extend to scenarios where labeled examples are scarce or novel invisible domains. The proposed method is an improvement on a two-stage detector, which is the mainstream framework in FSOD tasks.

2.2. Few-Shot Learning

Few-shot learning solves an important problem frequently encountered in many real-world computer vision tasks. Many tasks have few (labeled) examples, and it is difficult to label these unlabeled examples. Few-shot learning makes use of prior knowledge to be quickly generalized to new tasks containing few labeled samples. Few-shot classification (FSC) is a classical task in the field of few-shot learning, which employs many types of methods to solve the few-shot problem. The most famous method is meta-learning, which learns using prior knowledge and experience to guide the learning of new tasks, with the condition that new tasks and training tasks must be in the same distribution. Methods based on meta-learning can be divided into three categories: metric-based, optimization-based, and model-based. The metric-based methods [20–23] are the most popular methods; they map support information and query information to one feature-embedding space and then judge the similarity of samples according to the score determined by similarity measurement. The Siamese network [20] proposed twin networks in which support examples and query examples are fed into each network. The twin networks share weights with

each other concurrently, and the distance between support and query features is modeled accurately by logistic regression. A prototypical network [21] embeds the support examples into the same space, finds the mean as a prototype representation for each class, and then classifies query examples using Euclidean distance to find the nearest class prototype. The matching network [22] encodes examples from support and query sets, and determines the cosine distance as a similarity score for judgment after applying the encoding information of the support sets to each piece of encoding information of the query set. The relation network [23] achieves relation scores after concatenating the features from support examples and each feature of the query examples. Optimization-based methods [29,30] allow models to learn good initialization parameters, which enables models to quickly adapt to new tasks using only a few support samples or after a few steps of gradient descent. Model-based methods [31] use external memory modules to save information extracted from support examples in order to help the model perform well in the case of few shots. Few-shot learning tasks are successfully performed in the field of classification, but are in their infancy in other areas of study, such as object detection. The rapid extension to object detection benefits from the success of FSC, particularly metric learning-based approaches.

2.3. Few-Shot Object Detection

Due to the rapid development of FSC tasks, few-shot object detection has begun to make continuous progress. Methods [20–23] based on metric learning in FSC are most widely used in FSOD tasks. Recent methods of FSOD tasks can be divided into three categories: meta-learning-based, parameter-based, and sample-processing-based methods. Meta-learning-based methods are the most common. Following the N-way K-shot strategy, which is widely used in FSC, these methods typically have two branches: a support branch and a query branch. Support examples and query examples are fed into these two respective branches, keeping the ratio as $N \times K:1$ and learning meta-knowledge through a task-based training strategy. The design of the networks causes query images to take support images as the dynamic condition to use to adapt to new tasks quickly. FSRW [24] was proposed based on YOLOv2, which extracts the important information regarding support images as reweighted vectors for each class and achieves reweighted features, using them to query images to obtain more representative features for prediction. Inspired by [24], Meta R-CNN [25] was proposed based on Faster R-CNN [3], which adds weight vectors as points of attention on each region of interest (RoI) feature; however, the weight vectors are fixed during the fine-tuning stage. Zhang et al. [27] proposed ONCE based on a center net and implemented incremental learning, which can detect novel classes directly by means of novel support examples without fine-tuning. FSDeView [26] is a more suitable fusion method based on Meta R-CNN [25], allowing the query branch to obtain more representative information from support images. From the perspective of the two key modules of Faster R-CNN [3], FSOD [32] proposed an attention RPN module, allowing the network to consider important support information when filtering foregrounds and backgrounds. Lee et al. [33] used two attention mechanisms based on multihead attention [34] which were easy to plug and play. Inspired by FSOD [32], the studies on DANa [35] and Meta Faster R-CNN [36] proposed related solutions to the problem of spatial relationships between support features and query features, and both achieved excellent performance. To generate high-quality proposals and make them discriminative, Zhang et al. [37] used support query mutual guidance and hybrid loss. In terms of enhancing the classifier and regressor, Li et al. [38] added a correction network in the support branch to refine the classification scores, and Huang et al. [39] proposed a dynamic classifier and semi-explicit regressor to improve the generalizability. Most networks are based on Faster R-CNN [3]; specifically, the methods in [40,41] were improved based on DETR [42] and ViT [43]. The parameter-based methods have simple structures compared with other methods. TFA [44] and MetaDet [45] divide the parameters of Faster R-CNN [3] into two parts (category-specific and category-agnostic) with their respective strategies. DeFRCN [46] was proposed from the perspective of solving the two key contradictions of Faster R-CNN [3]. The methods used in [47,48]

based on sample augmentation provide the support branch with richer features in terms of the scale and quantity of input, and those used in [49,50] process the potential relationship within the data to enhance the information on novel classes. Jiang et al. [51] enhanced the features for classification at the spatial, task, and regularization levels. Lu et al. [52] incorporated text-modal descriptions for each category to alleviate confusion regarding the classification of novel classes, and Chen et al. [53] employed category knowledge to guide parameter calibration. The approaches mentioned above used a similar training strategy, i.e., training on base classes to obtain a network with the ability to learn meta-knowledge, and then allowing it to learn the knowledge of novel classes after fine-tuning both novel and base classes. Previous research has explored various methods to use to optimize the utilization of support information from multiple aspects, leading to significant advancements in the development of a unified framework for few-shot object detection. However, previous studies have often overlooked the potential issues arising between the two branches. In contrast, our approach not only leveraged support information effectively from aspect of scale, but also allowed us to conduct an in-depth analysis to resolve the inherent contradictions between the two branches.

3. Proposed Methods

3.1. Problem Definition

We followed the primary settings of some existing networks [24–28] to perform the FSOD tasks. The entire task had two primary stages: meta-training and meta-testing. The primary function of the network was to obtain transferable class-specific and class-agnostic parameters through training in base classes with rich labeled samples in the meta-training stage and then to apply these parameters to the second stage, providing the network with the ability to effectively complete the object detection task in the case of few shots.

We divide the dataset into two parts: D_{base} and D_{novel} , where $D_{base} \cap D_{novel} = \emptyset$. C_{base} contained base classes in D_{base} with rich-labeled objects, and C_{novel} contained novel classes in D_{novel} , which simulated the situation with only a few annotated samples per class. For each sample $(x, y) \in (X, Y)$, x indicates an image, and $y = \{(cls_i, box_i) | i \in Obj_x\}$ indicates the class label and bounding box for the object in the image. In the meta-training stage, we used a multiple-episodes strategy and selected the support set $S = \{S_i\}_{i=1}^M$ and the query set $Q = \{Q_i\}_{i=1}^M$ from D_{base} randomly. To follow the N-way K-shot strategy, for each task $T_i = \{(S_1, \dots, S_N), I_i\}$, $S_i = \{S_i^1, \dots, S_i^K\}$ contained the support images for each class, selected from support set S for this task. One group of support images were positive, and the rest were negative in the support images. Typically, in [24–26], each support image $S_i^k = \{(s, mask_i) | s \in S\}$ contained an additional binary mask channel to determine which object was used in the image. The mask can only be assigned one of two values; the positions of the used bounding box were set to 1, and the remaining positions were set to 0. In this study, we processed the support data in advance; in each training step, there was only one object annotation for the corresponding class in a single image. In the meta-testing stage, we randomly selected the support and query sets from D_{novel} , which was similar to the meta-training phase. To strictly adhere to the setting of the N-way K-shot and to show the performance of the network with only N-labeled samples for each class, we selected only K images for each novel class and then one bounding box of the corresponding class in each image for fine-tuning.

Thus, the network used the training strategy of multiple episodes and N-way K-shot to simulate a few-shots condition by means of rich labeled samples. Thus, by learning the knowledge that could be generalized to the target domain with only a few labeled samples, the network focused on the features of the query images, but also dynamically considered the information from the support images to recognize novel objects.

3.2. Module Architecture

In this paper, we used Faster R-CNN [3], the most commonly used two-stage model, as a basic detector. We proposed a novel architecture that fully considered the relationship

between the support branch and query branch to learn generalizable knowledge. The entire architecture of the proposed method is shown in Figure 2. Specifically, we proposed the decoupled multi-scale attention (DMA-Net) method, which consists of two branches for receiving support images, including positive and negative support categories, and query images. The prototypes are computed from positive support images, which are the means of the K images for each class. We sent the query images, prototypes, and all support images into a weight-sharing backbone for feature extraction, and then sent the positive features and query features into DMAM to incorporate the information on the support images into the query image in a more detailed way. The positive and negative support features were then reshaped into attention vectors and fused with each ROI proposal. These two operations allowed the RPN to generate more accurate proposals and enabled the detector to perform classification and regression more effectively. Based on these operations, we constructed a novel network for the FSOD task. The details of each important module are introduced in Sections 3.2.1 and 3.2.2.

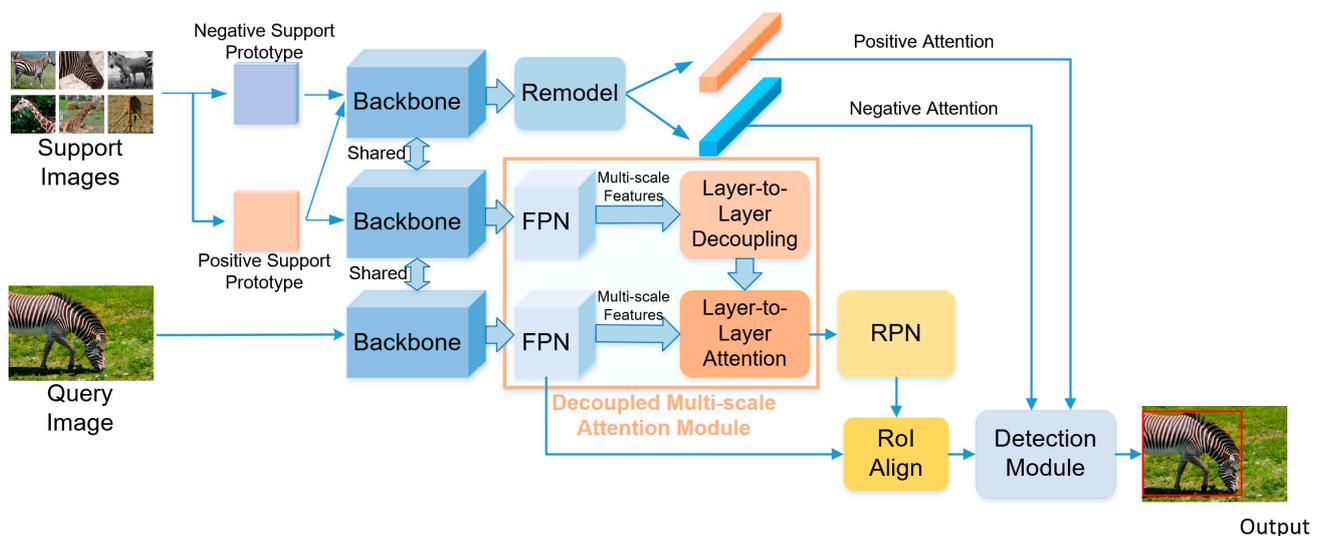


Figure 2. Architecture of DMA-Net.

3.2.1. Decoupled Multi-Scale Attention Module

How to manage the relationship between the support and query branches is the key point of FSOD networks based on a meta-learning strategy. In the previous methods, such as those in [25,26], support information was only used as the object of attention after RPN. Similarly, [51–53] utilized support information solely to enhance the classification ability after RPN, without considering the use of multi-scale or single-scale features. Although [47,52] extracted multi-scale features, they only added support information to the detection module without applying multi-scale feature extraction to the support branch, followed by multi-scale attention operation to fuse features from the two branches before RPN. Moreover, [28,32] simply performed a single-size fusion operation prior to RPN. They ignored the significance of the RPN module, which generates anchors and performs preliminary proposal screening, directly impacting subsequent classification performance. We also analyzed the contradiction difference between the proposed model and the original Faster R-CNN [3], and identified a problem of inconsistent parameter properties between the two branches. Therefore, we propose a novel module called the Decoupled Multi-scale Attention Module (DMAM), which takes into account the location and mode of support information fusion while ensuring the uniformity of the gradient and update parameters. Figure 3 consists of three primary parts, each of which will be described in detail below.

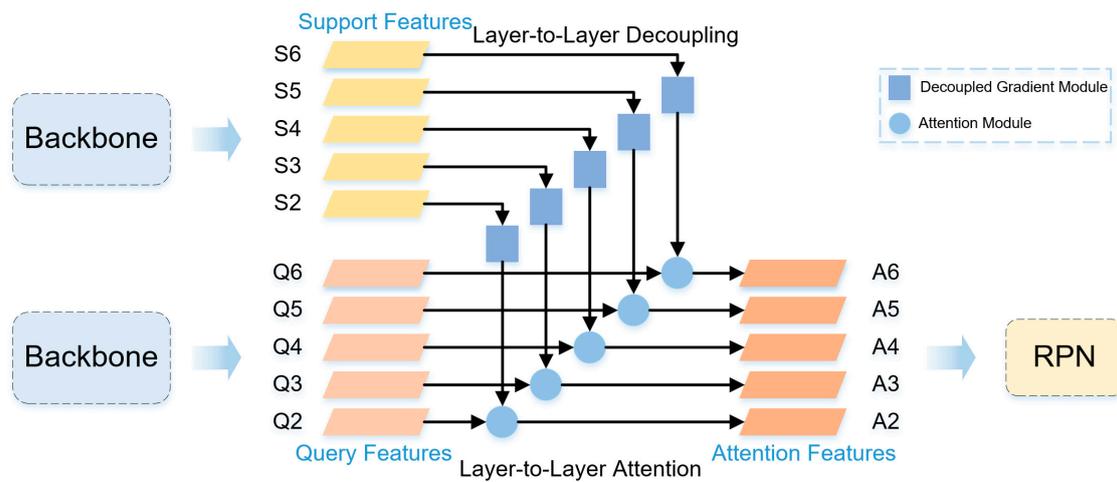


Figure 3. Decoupled Multi-scale Attention Module (DMAM).

We used a weight-shared network to perform feature extraction on both the query image $I_q \in \mathbb{R}^{H_q \times W_q \times C}$ and the support images $S = \{S_1, \dots, S_n\}_{n=1}^N$. In particular, the support features fed into DMAM came from the prototype of K positive support images $S_p = \{S_p^1, \dots, S_p^k\}_{k=1}^K$. The prototype was simply the mean of S_p , which can be defined as:

$$I_s = \frac{1}{K} \sum_{i=1}^K S_p^i, S_p^i \in S_p, I_s \in \mathbb{R}^{H_s \times W_s \times C}, \tag{1}$$

We fed the support image I_s and query image I_q into ResNet50 to extract corresponding preliminary features f_s and f_q . Subsequently, we applied a fusion operation to combine the support information with the query features before the RPN, allowing the support features to participate in proposal generation. Therefore, the RPN had a stronger ability to filter anchors based on the foreground and background, which reduced the burden of passing many irrelevant objects to the subsequent tasks of the detector and performed better when using the detector with the help of support information. The processing of the features which were input into the RPN consisted of two parts, feature extraction and feature fusion, which will be introduced next.

(1) Feature Extractor

The FSOD task is still an object detection task in essence, and faces the problem of recognizing small objects, which becomes more serious in the case of rare samples. Training with multi-scale features is an important operation that can ameliorate the problem of small objects and enrich available feature information as much as possible when the sample information is sparse. We use a feature pyramid network (FPN) [19] to extract multi-scale features. After a bottom-up pathway in Resnet50, we implemented the outputs of the last residual blocks in each stage and denoted the outputs of conv2, conv3, conv4, and conv5 as {C2, C3, C4, C5}. After the set of feature maps had been determined, a top-down pathway was applied. As shown in Figure 4, we used a 1×1 convolutional layer to perform channel dimension alignment, and the feature dimension was set to $d = 256$ in this paper. Subsequently, we applied nearest-neighbor upsampling to upscale the features of the upper layers and merged these upsampled features with the channel dimension-aligned feature maps in the corresponding layers using elementwise addition. In this iteration, we obtained P5 after simply reducing the channel dimension, and P6 by performing a max-pooling on P5. We also used a 3×3 convolutional layer after each merged feature map to mitigate the effect of upsampling. The final set of features {P2, P3, P4, P5, P6} was obtained by the above operations. If we were to send this feature set into the RPN for proposal generation directly, the RPN would lose contact with support information. It is critical to establish a relationship with support information and to do so prior to the RPN. It is also convenient

to use an attention operation. Constructing support features for use as weights and acting on the query features gives the positions related to objects high weight, and effectively transfers the attention of the RPN to a position related to the objects. In [28,36], single-scale support features were used to transfer the attention, which is inappropriate in the case of multi-scale query features, and many problems regarding feature size must be considered. According to the above description, we applied an FPN after positive support features to extract multi-scale support features {S2, S3, S4, S5, S6}. At all levels of the FPN, shallow features were rich in space information and deep features were rich in semantic information; thus, we used both shallow features and deep features of the support images. We also considered spatial information and semantic information together, which can alleviate the problems surrounding spatial information in FSOD tasks. After extracting the features of the positive support images and query images, we performed an attention operation layer to layer to achieve effective fusion and to maximize the use of support information.

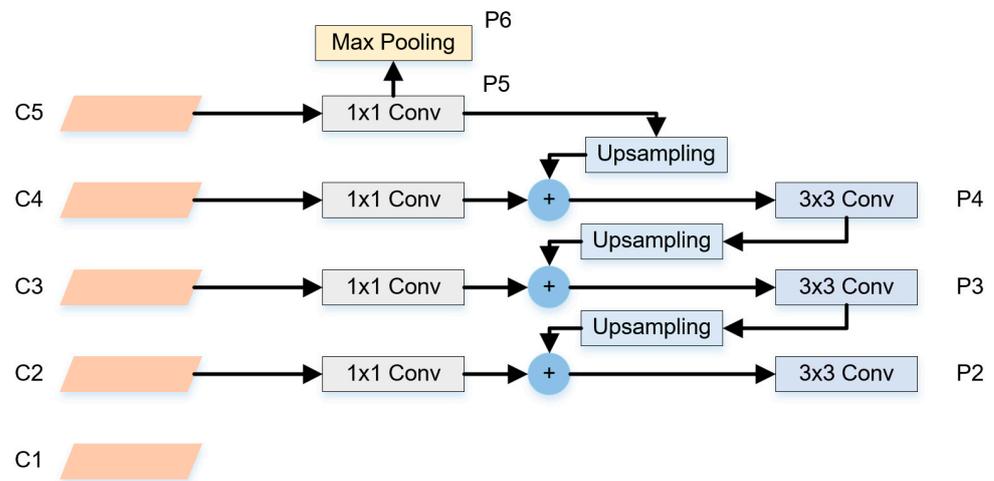


Figure 4. Multi-scale Feature Extractor.

(2) Attention Module

In this paper, we used a simple attention module to construct the relation between the query and support branch prior to the use of the RPN. We added FPN to both the positive support branch and the query branch, and there were five layers of features with different scales in each branch. Therefore, we were obligated to add attention layer by layer. Constructing a relation scale to scale can also reduce the influence between different scale features. There were five attention modules in five layers, which represented the same feature fusion method but had different feature sizes. As shown in Figure 5, query features and positive support features were fed into the attention module in each layer. We denoted query features as $Q \in t^{H \times W \times C}$ and positive support features as $S \in t^{M \times M \times C}$. We fused them using a depth-wise convolution method, processed the positive support features as a convolution kernel, and then used this kernel to perform sliding computing on the query features. The final attention feature map was obtained after this operation. The depth-wise convolution can be defined as:

$$A_{h,w,c} = \sum_{i,j} S_{i,j,c} \cdot Q_{h+i-1,w+j-1,c}, i, j \in \{1, \dots, M\}, \tag{2}$$

where A is the final attention feature, S and Q represent features from the support branch and the query branch respectively. In this study, we used global average pooling on positive support features, which is defined as:

$$S'_{1,1,c} = \frac{1}{M \times M} \sum_{h,w} S_{h,w,c}, h, w \in \{1, \dots, M\}, \tag{3}$$

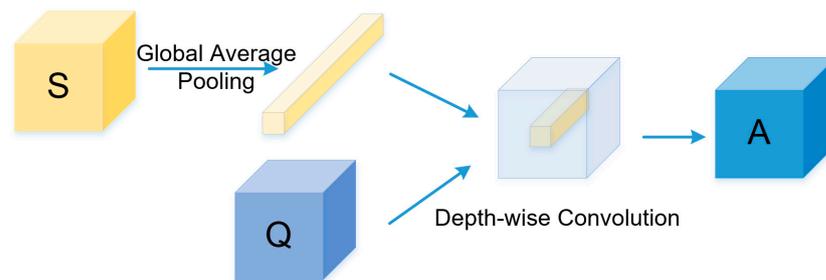


Figure 5. Attention module.

Therefore, the kernel size generated by $S \in t^{S \times S \times C}$ was processed to $1 \times 1 \times C$. Using 1×1 convolution can reduce the burden of the parameters, spatial clutter, and the consumption of query information itself. We used one 3×3 conv and two sibling 1×1 convs on the final attention feature to generate proposals for each corresponding layer. For $\{A2, A3, A4, A5, A6\}$ at each layer, we assigned the anchors with areas of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ pixels, and all used the multiple aspects ratios of $\{1 : 2, 1 : 1, 2 : 1\}$, similarly to [19]. We also obtained 15 anchors in total. At the stage of filtering anchors, we reserved anchors as positive and negative proposals based on their intersection-over-union (IoU) ratios with a ground-truth bounding box. The strategy was similar to that used in [3]; we set the anchor with an IoU higher than 0.7 or with the highest IoU and the corresponding ground-truth bounding box as positive proposals and assigned the anchor with an IoU below 0.3 to all ground-truth bounding boxes to negative proposals. Based on the above operations, we were able to obtain an RPN with the capability to reserve more proposals related to the object than before.

(3) Decoupled Gradient Module

In this paper, we conducted an analysis of the discrepancies between the proposed model and the original Faster R-CNN [3], and proposed decoupling between the support branch and the query branch to establish consistency in the type of gradient and in the updated parameters, thereby reducing the mutual interference between the two branches.

The main contradictions of Faster R-CNN [3] are as follows. It consists of three primary modules for high-quality object detection tasks: RPN, RCNN, and a backbone shared by RPN and RCNN. The RPN module generated anchors through the features extracted by the backbone and filtered out proposals sent to the RCNN module for detection. Concurrently, the RPN screened out anchors as positive samples as well as negative samples that were used to calculate the loss to optimize the RPN. This process was class-agnostic, because the RPN filter proposals based on the foreground and background did not involve proposal categories. The RCNN module first mapped the proposals from the RPN to the features extracted by the backbone, and then produced a series of proposals of the same size for the detection module through the RoI alignment operation for further classification and bounding box regression of the objects. This process was class-specific, because the RCNN was obliged to perform classification for each proposal in order to find the categories corresponding to their class scores. According to the above details, the RPN and RoI alignment module used the features extracted by the backbone concurrently, but their operating process was different (class-agnostic and class-specific, respectively), as were their purposes: one told the network where to look, while the other told the network what to look at. However, Faster R-CNN [3] was an end-to end system that optimized all the modules jointly through the following objective function:

$$\mathcal{L}_{\text{total}} = \underbrace{(\mathcal{L}_{rpn}^{cls} + \mathcal{L}_{rpn}^{reg})}_{rpn\text{task}} + \eta \cdot \underbrace{(\mathcal{L}_{rcnn}^{cls} + \mathcal{L}_{rcnn}^{reg})}_{rcnn\text{task}}, \quad (4)$$

where η is a balanced hyperparameter for the *rpn* and *rcnn* tasks. Two detection tasks in the RCNN have contradictions in the joint optimization process (i.e., task classification requires

translation-invariant features, while task regression requires translation-covariant features). Therefore, there may be potential optimization problems if these contradictions are used in the scenarios of rare samples.

To solve the contradictions of RPN, [46] used the decoupling module to decouple the RPN, RCNN, and backbone. An affine function was used during forward propagation so that the RPN and RCNN would not directly share features from the backbone. In the backward propagation stage, there was a decoupling coefficient multiplied by the gradient, which weighted the gradients from the RPN and RCNN. Typically, the decoupling coefficient of the RPN is set near 0, and the gradient of the RCNN is given a high weight so that the gradient of the RCNN leads the update of backbone parameters, while the gradient from the RPN loss is used to optimize the module itself. To resolve the contradictions between the classification and regression tasks in the detection module, the two tasks were separated to construct two branches, allowing the support branch to participate in the classification task. Thus, the root cause of the two primary contradictions directly shared the same features, leading to the potential optimization problem.

However, the proposed method avoided the aforementioned potential problems, as the RPN and RCNN did not share the same features directly in our model. The features for the RPN were extracted by the feature extractor and then passed through the attention module, while the features for RoI were extracted by the original feature extractor. This operation avoids the contradictions between the RPN and RCNN, but also prevents the loss of information after the features pass through the attention module. We also did not directly use the same features for the classification and regression tasks. For the regression task, we directly used the features combined with positive support features, while the proposed method for the classification task involved both positive and negative support features. This will be described in Section 3.2.2. Therefore, no decoupling operations were necessary for the forward propagation process.

Nevertheless, the support branch in our model intersected before the RPN and in RCNN. The RPN is a class-agnostic module that primarily detects object locations, while the RCNN is a class-specific module that performs object classification. The support branch in our model was also class-specific. Therefore, there may have been a similar contradiction of Faster R-CNN between our support and query branches if the type of the gradient of RPN and that of the support branch were inconsistent. To prevent the contradictions, we added a decoupled gradient module (DGM) between the support and query branches before the RPN, and their gradients were decoupled to reduce the possibility of unnecessary information exchange during the process of optimization, which may have caused the entire network to fail to reach the optimal solution. As shown in Figure 6, during the forward propagation process, the proposed module does not perform any operation because the proposed features are not directly shared during forward propagation. During backward propagation, we decoupled the gradient from the RPN. We set a decoupling coefficient λ , and the module obtained the gradient, multiplied it by λ , and propagated it to the support branch. The gradient descent step after applying DGM can be defined as:

$$\theta_s \leftarrow \theta_s - \gamma \cdot \mathbb{F} \left(\lambda \cdot \frac{\partial \mathcal{L}_{rpn}^q}{\partial \theta_{rpn}^s} \right) - \gamma \cdot \mathbb{R} \left(\frac{\partial \mathcal{L}_{rcnn}^q}{\partial \theta_{rcnn}^s} \right), \quad (5)$$

where θ_s , θ_{rpn}^s , and θ_{rcnn}^s are learnable parameters; γ is the learning rate; the two partial derivatives indicate the gradients from the RPN and RCNN; and \mathbb{F} and \mathbb{R} represent the gradient propagation of the FPN and remodeling parts in the support branch, respectively. During training, λ should be set to 0 to stop the gradient, and the gradient descent step becomes:

$$\theta_s \leftarrow \theta_s - \gamma \cdot \mathbb{R} \left(\frac{\partial \mathcal{L}_{rcnn}^q}{\partial \theta_{rcnn}^s} \right). \quad (6)$$

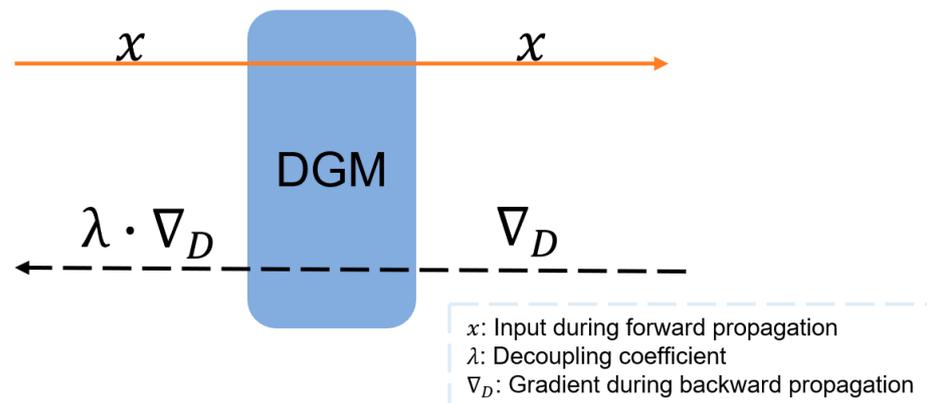


Figure 6. Decoupled gradient module.

When the parameters in the support branch are updated, the gradients are always class-specific, which avoids the unnecessary exchange between class-specific and class-agnostic information during the optimization process. We provide the pseudo code of DGM in Algorithm 1.

Algorithm 1: Decoupled Gradient Module, PyTorch-like

```
# ctx: context manager
# x: input during forward propagation
# _lambda: gradient decoupling coefficient
class DecoupleGradientModule(Function):
# feature forward
def forward(ctx, x, _lambda):
    ctx._lambda = _lambda
    return x
# feature backward
def backward(ctx, grad_output):
    grad_output = grad_output * ctx._lambda
    return grad_output, None
def decouple_layer(x, _lambda):
    return DecoupleGradientModule(x, _lambda)
```

3.2.2. Detection Module

In the detection module, we also added the support information for query features. The details of the detection module are shown in Figure 7. The entire module had three inputs: original features from the FPN, proposals from the RPN, and support attentions from the support branch. We sent all support images to the backbone for feature extraction and remodeled these features into attention. Specifically, we used Res1-3 blocks of ResNet50 for feature extraction and Res4 blocks for preliminary remodeling, we used global average pooling on support features to obtain remodeled features $\mathcal{F}_s \in \mathbb{R}^{1 \times 1 \times 2048}$, and the attention vectors of positive support features and negative support features were obtained using sigmoid on \mathcal{F}_s . We sent the proposals into the RoI alignment module for feature mapping and performed the RoI alignment operation to obtain a series of features of the same size, which was convenient for the subsequent classification and regression tasks. Specifically, we mapped the proposals to the original features generated by the FPN rather than the features that were added with the support information that was sent to the RPN. This operation reduced the loss of query information in the proposals. Because the subsequent detection tasks had contradictions mentioned previously, we processed the proposals and sent them into two parallel heads for box classification and box regression. For the box regression task, we fused positive support attentions with proposals using channel-wise multiplication, and then performed smooth L_1 on them for regression loss. For box classification, we fused

the proposals with positive support attentions and negative support attentions and used a matching method [35] on them for classification, enabling the network to recognize objects of the same class and distinguish objects of different classes simultaneously. The input of the support branch was 2-way 3-shot in the proposed experiments; thus, we obtained a query image I_q ; positive support images I_p^c , which have objects of the same class c ; and negative support images I_p^n , which have objects of different classes n . Positive proposals P_p and negative proposals P_n were generated after attention fusion. For each task, we filtered out 512 proposals from the RPN module, and the ratio of positive samples to negative samples was 1 : 3. P_p was composed of P_p^{fs} and P_p^{bs} , and P_n was composed of P_n^{fs} and P_n^{bs} . We set the label of foreground $P^{fs} = \{P_p^{fs}, P_n^{fs}\}$ to 1 and the rest to 0. We sorted background proposals $P^{bs} = \{P_p^{bs}, P_n^{bs}\}$ in descending order based on their class scores. According to the ratio of P^{fs} , P_p^{bs} , and P_n^{bs} of 1 : 2 : 1, proposals were selected to reconstruct the positive samples and negative samples that were used to calculate class loss. The negative samples contained the negative proposals wrongly divided into the foreground, which can help the network to distinguish objects of different categories when matching objects of the same class. The loss of each image is defined as $L = L_{matching} + L_{box}$, and the total loss is defined as:

$$L_{total} = \underbrace{(L_{cls} + L_{box})}_{L_{rpn}} + \eta \cdot \underbrace{(L_{cls} + L_{box})}_{L_{rcnn}}, \tag{7}$$

which is the same as that in [3], except that the L_{cls} in L_{rcnn} is $L_{matching}$ in this paper, the matching loss is the cross-entropy, and η is a balancing parameter. The pseudo-code of the proposed method is shown in Algorithm 2.

Algorithm 2: The pseudo-code of the proposed method.

Input: Base set D_{base} , support images I_s , query image I_q , initialized RPN parameters θ_{rpn} , RCNN parameters θ_{rcnn} , and support branch parameters θ_{sup} .

Output: Updated RPN parameters θ_{rpn} , RCNN parameters θ_{rcnn} , and support branch parameters θ_{sup} .

- 1: **for** $I_q \in D_{base}$ **do**
 - 2: Calculate the prototypes P_{ps}, P_{ns} of positive support images I_{ps} and negative support images I_{ns} ;
 - 3: Extract and fuse the multi-scale features f_q, f_{ps} of query image I_q and positive prototypes P_{ps} ;
 - 4: Extract the region of the interest features \hat{f}_q with the RPN and RoIAlignment;
 - 5: Extract single-scale features f_s of support prototypes P_s and remodel the support features as attentions A_s ;
 - 6: Fuse the region of the interest features \hat{f}_q with attentions A_s ;
 - 7: Calculate loss L_{total} with a detector according to Equation (7);
 - 8: Update θ_{rpn} to minimize the training loss L_{rpn} and update θ_{rcnn} and θ_{sup} to minimize the training loss L_{rcnn} ;
 - 9: **end for**
-

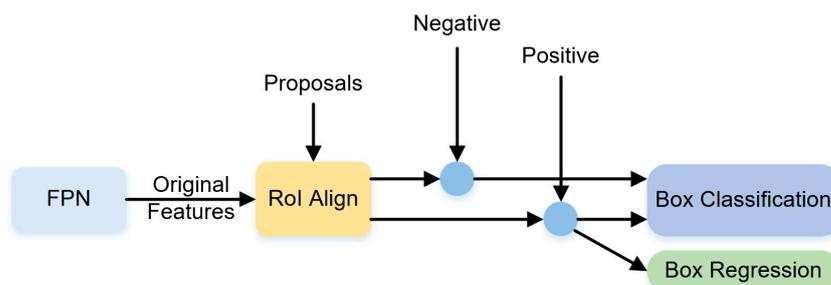


Figure 7. Detection module.

4. Experiments and Discussions

4.1. Implementation Details

In this paper, we used the ResNet50 as a preliminary feature extractor. Specifically, we used Res1–3 blocks for feature extraction in the first support branch in Figure 2 and Res4 blocks for preliminary remodeling. The experiments were all implemented in PyTorch [54] on a workstation with an NVIDIA 2080Ti. The shorter sides of the query images were resized to 600 pixels, while the longer sides were cropped to 1000 pixels. The support images were cropped around the target object, zero-padded, and then resized to obtain square images with dimensions of 320×320 . We applied SGD to optimize the model with a batch size of 4, a momentum of 0.9, and a weight decay of 0.00001. Subsequently, we trained the network with a learning rate of 0.001 for 7 epochs, and then decreased it to 0.0001 at epoch 8. The anchors were assigned scales of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$, and the three aspect ratios were $\{1 : 2, 1 : 1, 2 : 1\}$, while the decoupling coefficient λ was 0.01 in DMAM.

4.2. Experimental Settings

4.2.1. Datasets

In the proposed experiments, we used MSCOCO2017 for training and evaluation. COCO is the most widely used and challenging dataset of the previous FSOD benchmarks, and contains 80 categories in the object detection task, 11.8 K images in the training set, and 5 K images in the validation set. We took the validation data as the testing data, as in previous studies. We considered 20 categories that coincided with PASCAL VOC as novel classes, and the remaining 60 categories that did not belong to PASCAL VOC as base classes.

4.2.2. Training

Following the N-way K-shot training strategy, we set $N = 2$, $K = 3$ for each episode in all experiments. For each episode, there was one query image and two-way support sets: one way contained positive support images and the other way contained negative support images; each way had three support images. During the fine-tuning process, we preprocessed the support images such that each image in the input of the support branch contained exactly one object of the corresponding class. Different from the processing in [24–26], a binary mask was added as the fourth channel of the support images to indicate the support object for the network during training. Our processing avoided the feature extraction module needing two heads due to the inconsistency of the channels when extracting features of support and query branches. The annotations of the query objects and corresponding support objects that belonged to novel classes were all moved, and there was no overlap between novel and base data.

4.2.3. Evaluation

Following the general few-shot paradigm in previous studies, we intended to prepare K images of each novel class as support images during testing. However, previous studies used different settings for fine-tuning and evaluation. During fine-tuning, the query image may have several categories with many objects; thus, the definition of K-shot would be different. In [24], the K-shot was defined as having K bounding boxes per class, while [55] defined the K-shot as having K images. In the evaluation phase, [24–26] only used the query image as the input and the fixed attention vectors obtained by the support branch in the fine-tuning phase as support information. In [32,35], K novel support pools were used to provide information for the query image; thus, there was more ambiguity in comparison. To implement the N-way K-shot paradigm and to thoroughly evaluate the proposed network, we followed the protocol outlined in [35] and compared the results with studies that used the same evaluation settings. In addition, we conducted incremental experiments without any samples of novel classes to fine-tune the proposed model, and we used the network trained on base classes to evaluate the test data directly. Because the purpose of the FSOD

task was to solve the problem of rare samples, we hoped that the proposed model would be able to recognize novel classes using only K bounding boxes of novel classes, without using additional samples and spending additional time fine-tuning. We also hoped that the novel class would be detected at any time it was registered.

4.3. Generic FSOD Protocol

In Table 1, we report the results of the DMA-Net on novel classes of the COCO dataset. We followed the evaluation method widely used in previous papers and compared the proposed results with those reported in previous papers. As mentioned previously, we used base classes to simulate the few-shot situation for episode training and then fine-tuned the unseen novel classes. To test the generalizability of the proposed model, we only selected 10/30 bounding boxes for each novel class for fine-tuning in order to address the condition that there were only K samples for us to use to help the model recognize the novel objects. In this challenging setting, the proposed results of AP and AP₇₅ in 10/30 shots were better than those of many existing studies. The DMA-Net was only trained for 8 epochs, and this performance could be achieved after fine-tuning for 4 epochs. The training took 2 days, 18 h, and 56 min, while DAnA [35] needed 4 days, 1 h, and 42 min; thus, it can be concluded that the DMA-Net can save considerable training and fine-tuning time and still exhibit a good performance. In Figure 8, we show the convergence curve of the loss function in Equation (7).

Table 1. Fine-tuning results on novel classes of COCO. “-”: no reported results.

Method	10 Shot		30 Shot	
	AP	AP ₇₅	AP	AP ₇₅
Feature Reweighting [24]	5.6	4.6	9.1	7.6
Meta R-CNN [25]	8.7	6.6	12.4	10.8
MPSR [47]	9.8	9.7	14.1	14.2
TFA w/cos [44]	10.0	9.3	13.7	13.4
Attention RPN [32]	11.1	10.6	-	-
Meta Faster R-CNN [36]	11.3	9.8	15.9	14.7
FSDetView [26]	12.5	9.8	14.7	12.2
FSDetView + PsP [33]	13.4	9.1	17.1	14.7
CME [56]	15.1	16.4	16.9	17.8
DMA-Net	17.2	15.3	18.6	17.2

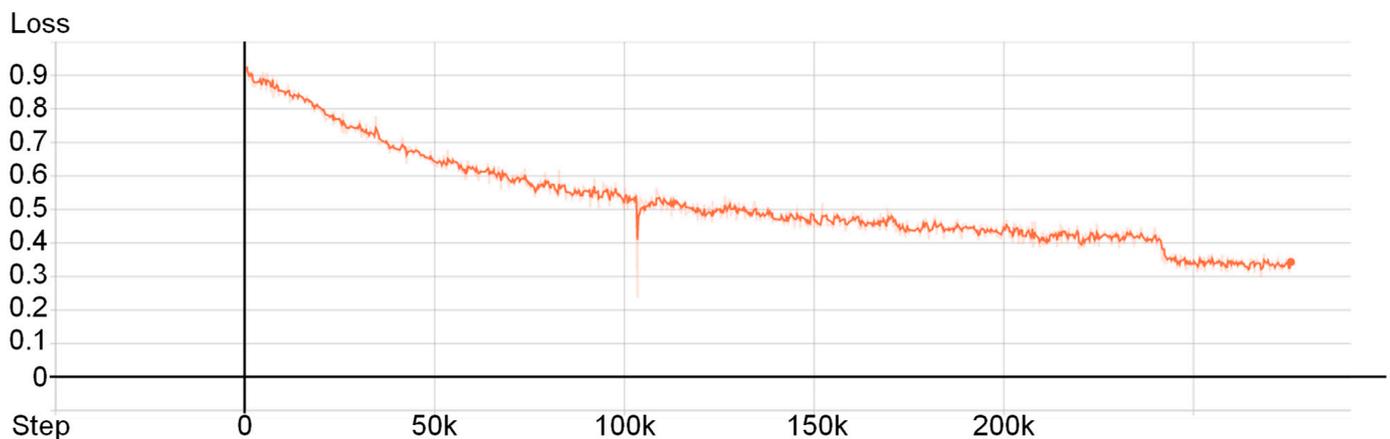


Figure 8. The convergence curve of the loss function in Equation (7).

4.4. Incremental FSOD Protocol

In this section, we describe our evaluation of the DMA-Net in a more challenging situation. We directly tested the proposed model on novel classes after the network was trained on base classes without any novel samples for fine-tuning. Therefore, we only needed to prepare K bounding boxes of each novel class as support data to recognize objects of that class. In this setting, the network was able to recognize unseen novel classes at any time, which is referred to as incremental FSOD. In Table 2, the studies that we compared, including the proposed method, the data input, and result evaluation, showed the same result, and there was no ambiguity in comparison. Table 2 shows that the DMA-Net outperformed the SOTA method DAnA-FasterRCNN [35] by 0.5/0.8/0.1 mAP according to the AP/AP₅₀/AP₇₅ metrics. An excellent FSOD network should also achieve good performance with base classes. Table 2 shows a comparison between FSOD methods and the original Faster R-CNN in terms of accuracy with base classes [3]. FGN [57] and attention RPN had a large gap with Faster R-CNN [3], while the performance of the DMA-Net is nearer to that of Faster R-CNN [3]. Therefore, the proposed network is able improve accuracy with novel classes without losing accuracy with base classes, and DMA-Net can achieve a good performance on AP/AP₅₀/AP₇₅ metrics of both novel and base categories. In addition, DAnA [35] was trained for 16 epochs on COCO datasets, while the proposed network had fewer parameters and only required half of the epochs to reach SOTA accuracy; thus, DMA-Net also reduces additional costs compared with the methods used in previous studies. DAnA [35] evaluated models with different numbers of support images under 2-way 3-shot training, including the case of 1 shot and 5 shots. In Table 3, we show the incremental results for the novel and base classes of COCO with different numbers of support images. The proposed model performed best in the case of 1 shot, as well as the case of 3 shots, but performed less well in the case of 5 shots. The proposed model improved markedly with a decrease in the number of support images, which demonstrates that DMA-Net is more suitable for the few-shot case, but loses its generalizability as the number of support images increases. This result likely occurs because we only chose the average features of their prototypes when processing support images; therefore, how to better extract the features of support images should be investigated in more detail.

Table 2. Incremental results for novel and base classes of COCO. Faster R-CNN [3] is trained and evaluated under generic object detection protocol, so the results on novel categories are not applicable.

Method	Novel Categories			Base Categories			Parameters
	AP	AP ₅₀	AP ₇₅	AP	AP ₅₀	AP ₇₅	
Faster R-CNN [3]	N/A	N/A	N/A	34.3	58.3	35.6	4.76×10^7
Meta R-CNN [25]	11.1	25.3	8.5	28.6	52.5	28.4	4.76×10^7
FGN [57]	10.5	22.5	8.8	25.5	46.4	25.5	1.48×10^8
Attention RPN [32]	10.1	23.0	8.3	22.4	40.8	22.2	1.03×10^8
DAnA-FasterRCNN [35]	14.0	28.9	13.0	29.4	50.6	30.3	1.42×10^8
DMA-Net	14.5	29.7	13.1	30.1	53.0	31.2	6.59×10^7

Table 3. Incremental results for novel and base classes of COCO with different numbers of support images.

Method	Novel Categories						Base Categories					
	AP		AP ₅₀		AP ₇₅		AP		AP ₅₀		AP ₇₅	
Given Supports	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
Meta R-CNN [25]	8.7	11.2	19.9	25.9	6.8	8.6	27.3	28.5	50.4	52.3	27.3	28.2
FGN [57]	8.0	10.9	17.3	24.0	6.9	9.0	24.7	26.9	44.3	47.6	25.0	27.4
Attention RPN [32]	8.7	10.6	19.8	24.4	7.0	8.3	20.6	23.0	37.2	42.0	20.5	22.4
DAnA-FasterRCNN [35]	11.9	14.4	25.6	30.4	10.4	13.0	27.8	32.0	46.3	54.1	27.7	32.9
DMA-Net	12.7	14.0	26.6	29.4	11.3	12.5	28.0	30.2	49.8	53.3	29.3	31.1

4.5. Ablations

We conducted an ablation study in a situation without fine-tuning and analyzed the effect of each module.

4.5.1. Effectiveness of Layer-to-Layer Attention

The few shots of the novel class make the information of this class valuable; therefore, we hope to make the best use of the information about these novel objects when recognizing objects. Using the features of the query image with different scales can improve the use of query information. We followed DAnA [35] to evaluate the proposed model on the COCO benchmark, in which there are many small objects. It is also a critical point for FSOD models to improve their ability to detect small objects; thus, we applied a feature pyramid in the query branch for multi-scale feature extraction. Ablation (a,b) in Table 4 shows that multi-scale feature extraction is more effective than single-feature extraction. In Figure 9, the visualization of the multi-scale features before RPN is shown. As the level of feature layers increased, attention was shifted from small objects to large objects, and we obtained the common effect of these layers concurrently using multi-scale features. Therefore, when we discovered large, easy-to-detect objects, we were also able to consider detecting small objects that were more difficult to detect. As shown in Figure 10, the model using multi-scale features for detection had a stronger recognition ability for small objects.

Table 4. Effectiveness of each module.

	Multi-Scale Feature	Scale-to-Scale Attention	Decoupled Gradient	AP	AP ₅₀	AP ₇₅
a				11.1	24.6	8.5
b	✓			11.7	24.7	10.3
c	✓	✓		12.5	26.9	10.9
d	✓	✓	✓	14.5	29.7	13.1

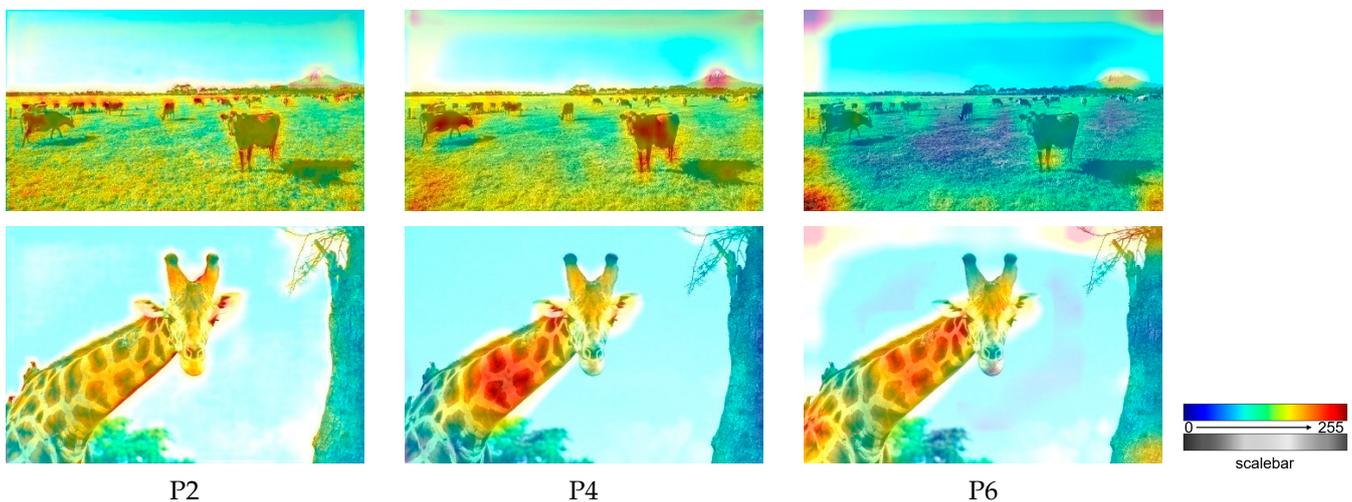


Figure 9. Visualization of the multi-scale features. The deeper the color is, the more the model responded. Each column represents the feature at level P2, P4, and P6, respectively.



Figure 10. Incremental results regarding novel classes of COCO with 3 support images. The first row shows the detection results using multi-scale features, and the second row shows the detection results using single-scale features.

Adding support information into query features is a key point in FSOD models. The models in [24,25] did not fuse support information with query features before RPN, and the models in [35,36] utilized complex fusion methods before RPN. In this study, we would have created two problems if we had added a fusion operation before RPN. First, the features of the query image are at different scales, while the features of support images are single-scale, and there is a scale problem when fusing features. We, thus, used the feature pyramid operation in the support branch to obtain features with scales corresponding to the query features and reduced the loss of information caused by direct scaling of the features. This facilitated the subsequent scale-to-scale feature fusion for features of the two branches. Second, the introduction of a feature pyramid increases the complexity of the model. Due to the rarity of novel classes, the complex fusion operations lead to overfitting of the model. In addition, the model is trained on base classes and evaluated on novel classes directly without fine-tuning, which causes the model to fit the base classes well and lose its generalizability for novel classes. We used the fusion operation described in Section 3.2.1-(1) layer to layer. Ablation, shown in (a,c) of Table 4, confirmed the effectiveness of this module. As shown in Table 5, we performed an ablation study of the feature fusion operation of two branches before RPN and determined what fusion operation would be used and where (Line e in Table 5 is the proposed final model result). First, the ablation result (a,e) in Table 5 shows that the fusion operation involved in the upsampling (in Figure 4) of query feature extraction allows the model to obtain better results. Second, we compared a simple fusion operation (Section 3.2.1-(2)) and a complex fusion operation (Channel attention module [58]), as shown in Table 5 (b,e). The complex fusion operation did not cause the model to perform better, and the result c in Table 4 and b in Table 5 show that complex operations can even reduce the ability of the model to fit novel classes. To understand the effect of the proposed module in more detail, in Figure 11, we visualize the features for detection. We chose feature P5 of the proposed multi-scale features to compare with the single feature in the model without DMAM. With the proposed DMAM, the model paid more attention to the objects belonging to the novel class, and the attention was more focused.

Table 5. Ablation study of the primary modules.

	Fusion after Upsampling	Fusion before Upsampling	Complex Fusion	Simple Fusion	Category Agnostic Fusion	DGM before RPN	DGM after RPN	AP	AP ₅₀	AP ₇₅
a		✓		✓		✓		12.9	27.0	11.3
b	✓		✓			✓		12.4	26.2	10.8
c	✓				✓	✓		13.4	27.9	11.6
d	✓			✓		✓	✓	13.3	28.1	11.7
e	✓			✓		✓		14.5	29.7	13.1

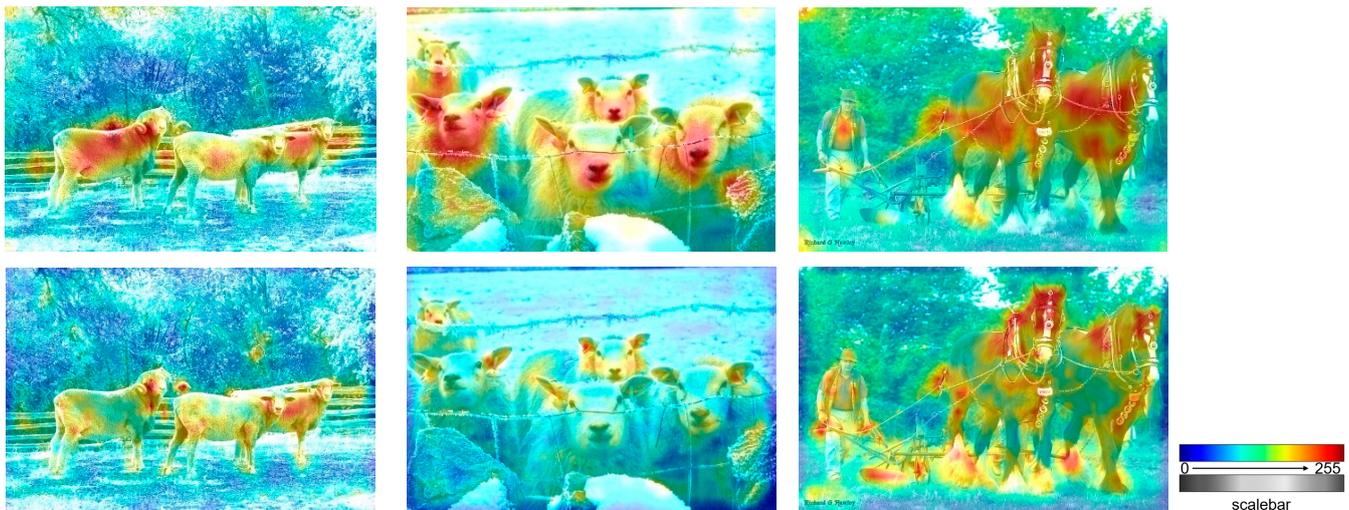


Figure 11. Visualization of the features for detection. The deeper the color is, the more the model responded. The first row shows the features of the model with DMAM, and the second row shows the features of the model without DMAM. For a fair comparison between the multi-scale features, we selected feature P5, with the same size as the model without DMAM.

The input of the support branch includes positive support samples and negative support samples; both belong to the base classes during training and to the novel classes during evaluation. Prior to RPN, we incorporated both positive and negative support features as attention mechanisms to query features, so that class-agnostic model RPN would be able to obtain more class-agnostic information. However, the ablation result (c,e) in Table 5 demonstrates that only fusing positive support information prior to RPN yielded better performance. How to fuse the category-agnostic information effectively is also worthy of further study.

4.5.2. Effectiveness of Decoupled Gradient

As described in Section 3.2.1-(3), there are contradictions in Faster R-CNN [3], although these contradictions are not relevant. However, the proposed model has one more support branch than Faster R-CNN [3], and there is a significant amount of information exchanged between the two branches. To reduce the unnecessary exchange of optimization information between the support and query branches, we used a DGM after the attention operation in the support branch and before the RPN in the query branch to decouple the gradient, which allowed the class-specific gradient of the network to update the class-specific parameters of the support branch. Ablation (c,d), as in Table 4, shows the effectiveness of the DGM. We also compared the addition of DGM to the detection module, and the ablation result (d,e) in Table 5 shows that the proposed model did not require additional decoupling operations.

4.6. Discussion

Based on the experimental results, the proposed network achieved comparable results for the base and novel classes in the COCO dataset with widely used baselines. Table 2 shows that the proposed network achieved a good performance on both the novel and base classes without fine-tuning. We provide many visualizations of the detection results of novel classes in the COCO dataset in Figure 12. We also used a multi-scale feature extractor in the query branch and integrated support information into the features before the RPN so that the RPN would be able to more clearly distinguish the foreground and background when rare samples are being evaluated. We proposed a layer-to-layer attention operation based on different scales to fuse the information between the support and query branch, which overcame the scale mismatch problem and avoided the overfitting caused by the increase in complexity. Gradient decoupling was performed between the two branches so that the property of the gradient and the parameters it updated could be unified. Regarding previous studies, only the support information was fused into the detection module in [25]; the support information was fused before the RPN with a single size in [28,36]; and the gradient between the backbone, RPN, and RCNN was decoupled in [46]; on the other hand, our method effectively utilized the support information from the aspect of scale and determined the specific location and mode of fusion operation. In addition, we analyzed the contradicting difference between the proposed model and original Faster R-CNN [3] in Section 3.2.1-(3). We were obliged to decouple only the gradient between the support branch and the RPN, and the experimental results demonstrate the effectiveness of the proposed methods; thus, we did not require additional decoupling operations. Our model not only achieved strong performance on novel classes, but also maintained a comparable level of detection accuracy to the original Faster R-CNN on base classes. Compared with the SOTA model DAnA [35] in the same setting, DMA-Net was able to achieve a better performance with fewer parameters and less training time without fine-tuning. However, in terms of detection efficiency, DMA-Net and DAnA [35] achieved FPS values of 7.28 and 9.45, respectively, in the same hardware environment. Although the gap was small, our model faced a disadvantage due to the utilization of multi-scale features during the extraction of information from two branches, leading to a significant increase in computational requirements. Therefore, reducing the computational burden of the proposed model should be a focus of future investigations. In addition, we have not yet solved the problem of how to use the proposed network to fuse category-agnostic information effectively by improving the extraction operation of the support features and the fusion operation between two branches. We will attempt to improve the proposed method in this regard in future work.

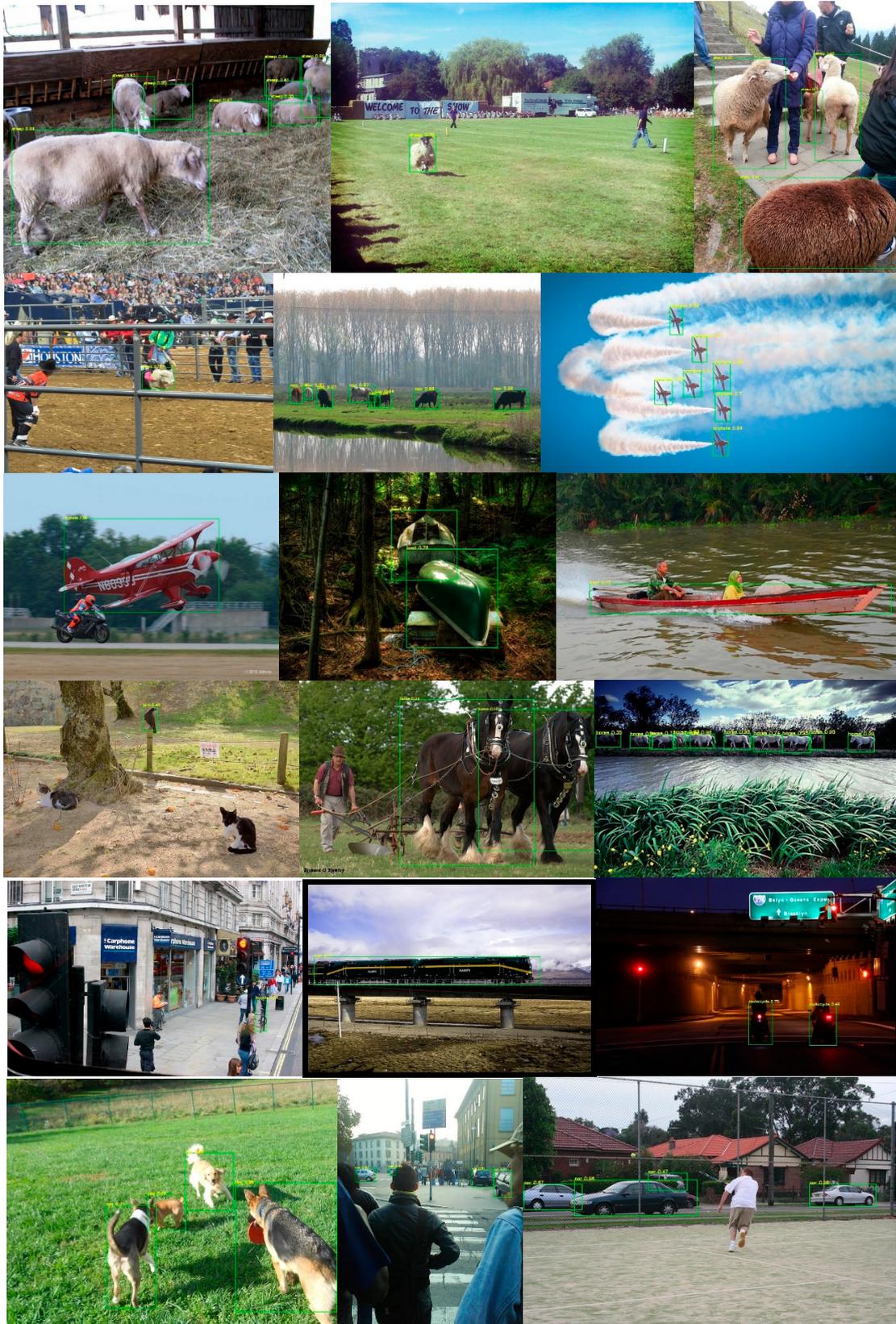


Figure 12. Visualization of detection results of novel classes in the COCO dataset (incremental results with 3 support images).

5. Conclusions

In this paper, we proposed a novel framework called Decoupled Multi-scale Attention (DMA-Net) to integrate support information into the features before the RPN, which consists of three parts: a multi-scale feature extractor, a multi-scale attention module, and a decoupled gradient module (DGM). With the cooperation of these modules, the proposed network achieved good results on the COCO dataset, particularly without fine-tuning, and DMA-Net achieved SOTA results on both the novel and base classes.

Author Contributions: Conceptualization, F.L. and Q.C.; methodology, X.X. and F.L.; software, X.X.; data curation, X.X.; validation, X.X.; writing—original draft preparation, X.X.; writing—review and editing, F.L. and Q.C.; supervision, F.L. and Q.C.; funding acquisition, Q.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by JSPS KAKENHI Grant Number 22K12079.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
2. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
3. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [[CrossRef](#)] [[PubMed](#)]
4. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as points. *arXiv* **2019**, arXiv:1904.07850.
5. Law, H.; Deng, J. CornerNet: Detecting objects as paired keypoints. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 734–750.
6. 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, 11–14 October 2016; pp. 21–37.
7. Cai, J.; Lee, F.; Yang, S.; Lin, C.; Chen, H.; Kotani, K.; Chen, Q. Pedestrian as points: An improved anchor-free method for center-based pedestrian detection. *IEEE Access* **2020**, *8*, 179666–179677. [[CrossRef](#)]
8. Zhu, L.; Lee, F.; Cai, J.; Yu, H.; Chen, Q. An improved feature pyramid network for object detection. *Neurocomputing* **2022**, *483*, 127–139. [[CrossRef](#)]
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
10. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
11. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
12. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
13. Cai, Z.; Vasconcelos, N. Cascade R-CNN: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6154–6162.
14. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
15. Miao, B.; Chen, Z.; Liu, H.; Zhang, A. A target re-identification method based on shot boundary object detection for single object tracking. *Appl. Sci.* **2023**, *13*, 6422. [[CrossRef](#)]
16. Chen, M.; Duan, Z.; Lan, Z.; Yi, S. Scene reconstruction algorithm for unstructured weak-texture regions based on stereo vision. *Appl. Sci.* **2023**, *13*, 6407. [[CrossRef](#)]
17. Xia, Q.; Lee, F.; Chen, Q. TCC-net: A two-stage training method with contradictory loss and co-teaching based on meta-learning for learning with noisy labels. *Inf. Sci.* **2023**, *639*, 119008. [[CrossRef](#)]
18. Wu, J.; Zhou, Y. An improved few-shot object detection via feature reweighting method for insulator identification. *Appl. Sci.* **2023**, *13*, 6301. [[CrossRef](#)]

19. Wang, Z.; Li, Y.; Chen, X.; Lim, S.N.; Torralba, A.; Zhao, H.; Wang, S. Detecting everything in the open world: Towards universal object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 11433–11443.
20. Koch, G.; Zemel, R.; Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In Proceedings of the ICML Deep Learning Workshop, Lille, France, 6–11 July 2015.
21. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. *Neural Inf. Process. Syst.* **2017**, *30*, 4080–4090.
22. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. Matching networks for one shot learning. *Neural Inf. Process. Syst.* **2016**, *29*, 3637–3645.
23. Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P.H.; Hospedales, T.M. Learning to compare: Relation network for few-shot learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1199–1208.
24. Kang, B.; Liu, Z.; Wang, X.; Yu, F.; Feng, J.; Darrell, T. Few-shot object detection via feature reweighting. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8419–8428.
25. Yan, X.; Chen, Z.; Xu, A.; Wang, X.; Liang, X.; Lin, L. Meta R-CNN: Towards general solver for instance-level low-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9577–9586.
26. Xiao, Y.; Marlet, R. Few-shot object detection and viewpoint estimation for objects in the wild. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 192–210.
27. Perez-Rua, J.M.; Zhu, X.; Hospedales, T.; Xiang, T. Incremental few-shot object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 13846–13855.
28. Wu, X.; Sahoo, D.; Hoi, S. Meta-RCNN: Meta learning for few-shot object detection. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 1679–1687.
29. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1126–1135.
30. Ravi, S.; Larochelle, H. Optimization as a model for few-shot learning. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
31. Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; Lillicrap, T. Meta-learning with memory-augmented neural networks. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1842–1850.
32. Fan, Q.; Zhuo, W.; Tang, C.K.; Tai, Y.W. Few-shot object detection with attention-RPN and multi-relation detector. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4013–4022.
33. Lee, H.; Lee, M.; Kwak, N. Few-shot object detection by attending to per-sample-prototype. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2022; pp. 2445–2454.
34. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
35. Chen, T.-I.; Liu, Y.-C.; Su, H.-T.; Chang, Y.-C.; Lin, Y.-H.; Yeh, J.-F.; Chen, W.-C.; Hsu, W.H. Dual-awareness attention for few-shot object detection. *IEEE Trans. Multimed.* **2021**, *25*, 291–301. [[CrossRef](#)]
36. Han, G.; Huang, S.; Ma, J.; He, Y.; Chang, S.-F. Meta faster R-CNN: Towards accurate few-shot object detection with attentive feature alignment. In Proceedings of the Conference on Artificial Intelligence, Online, 22 February–1 March 2022; pp. 780–789.
37. Zhang, L.; Zhou, S.; Guan, J.; Zhang, J. Accurate few-shot object detection with support-query mutual guidance and hybrid loss. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021; pp. 14424–14432.
38. Li, Y.; Zhu, H.; Cheng, Y.; Wang, W.; Teo, C.S.; Xiang, C.; Vadakkepat, P.; Lee, T.H. Few-shot object detection via classification refinement and distractor retreatment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021; pp. 15395–15403.
39. Huang, J.; Chen, F.; Huang, S.; Zhang, D. Instant response few-shot object detection with meta strategy and explicit localization inference. *arXiv* **2021**, arXiv:2110.13377.
40. Zhang, G.; Luo, Z.; Cui, K.; Lu, S. Meta-DETR: Few-shot object detection via unified image-level meta-learning. *arXiv* **2021**, arXiv:2103.11731.
41. Zhang, X.; Liu, F.; Peng, Z.; Guo, Z.; Wan, F.; Ji, X.; Ye, Q. Integral migrating pre-trained transformer encoder-decoders for visual object detection. *arXiv* **2022**, arXiv:2205.09613.
42. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable transformers for end-to-end object detection. *arXiv* **2020**, arXiv:2010.04159.
43. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16 × 16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
44. Wang, X.; Huang, T.E.; Darrell, T.; Gonzalez, J.E.; Yu, F. Frustratingly simple few-shot object detection. *arXiv* **2020**, arXiv:2003.06957.
45. Wang, Y.X.; Ramanan, D.; Hebert, M. Meta-learning to detect rare objects. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9925–9934.

46. Qiao, L.; Zhao, Y.; Li, Z.; Qiu, X.; Wu, J.; Zhang, C. DeFRCN: Decoupled faster R-CNN for few-shot object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 8681–8690.
47. Wu, J.; Liu, S.; Huang, D.; Wang, Y. Multi-scale positive sample refinement for few-shot object detection. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 456–472.
48. Xu, H.; Wang, X.; Shao, F.; Duan, B.; Zhang, P. Few-shot object detection via sample processing. *IEEE Access* **2021**, *8*, 29207–29221. [[CrossRef](#)]
49. Wu, A.; Han, Y.; Zhu, L.; Yang, Y. Universal-prototype enhancing for few-shot object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 9567–9576.
50. Zhu, C.; Chen, F.; Ahmed, U.; Shen, Z.; Savvides, M. Semantic relation reasoning for shot-stable few-shot object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021; pp. 8782–8791.
51. Jiang, X.; Li, Z.; Tian, M.; Liu, J.; Yi, S.; Miao, D. Few-shot object detection via improved classification features. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 2–7 January 2023; pp. 5386–5395.
52. Lu, X.; Sun, X.; Diao, W.; Mao, Y.; Li, J.; Zhang, Y.; Wang, P.; Fu, K. Few-shot object detection in aerial imagery guided by text-modal knowledge. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 1–19. [[CrossRef](#)]
53. Chen, C.; Yang, X.; Zhang, J.; Dong, B.; Xu, C. Category knowledge-guided parameter calibration for few-shot object detection. *IEEE Trans. Image Process.* **2023**, *32*, 1092–1107. [[CrossRef](#)] [[PubMed](#)]
54. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in PyTorch. In Proceedings of the 31st International Conference on Neural Information Processing System, Long Beach, CA, USA, 8 December 2017.
55. Chen, H.; Wang, Y.; Wang, G.; Qiao, Y. LSTD: A low-shot transfer detector for object detection. In Proceedings of the Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 2836–2843.
56. Li, B.; Yang, B.; Liu, C.; Liu, F.; Ji, R.; Ye, Q. Beyond max-margin: Class margin equilibrium for few-shot object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021; pp. 7363–7372.
57. Fan, Z.; Yu, J.; Liang, Z.; Ou, J.; Gao, C.; Xia, G.S.; Li, Y. FGN: Fully guided network for few-shot instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9172–9181.
58. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBAM: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 3–19.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.