Towards Resolving the Challenge of Long-tail Distribution in UAV Images for Object Detection

Weiping Yu*, Taojiannan Yang*, Chen Chen
University of North Carolina at Charlotte
{wyu4, tyang30, chen.chen}@uncc.edu

Abstract

Existing methods for object detection in UAV images ignored an important challenge – imbalanced class distribution in UAV images – which leads to poor performance on tail classes. We systematically investigate existing solutions to long-tail problems and unveil that re-balancing methods that are effective on natural image datasets cannot be trivially applied to UAV datasets. To this end, we rethink long-tailed object detection in UAV images and propose the Dual Sampler and Head detection Network (DSHNet), which is the first work that aims to resolve long-tail distribution in UAV images. The key components in DSHNet include Class-Biased Samplers (CBS) and Bilateral Box Heads (BBH), which are developed to cope with tail classes and head classes in a dual-path manner. Without bells and whistles, DSHNet significantly boosts the performance of tail classes on different detection frameworks. Moreover, DSHNet significantly outperforms base detectors and generic approaches for long-tail problems on VisDrone and UAVDT datasets. It achieves new state-of-the-art performance when combining with image cropping methods. Code is available at `https://github.com/we1pingyu/DSHNet`

1. Introduction

With the advance of unmanned aerial vehicles (UAVs), collecting high-quality images from the air has become convenient. Object detection plays a crucial role in many UAV applications, being a theme common to security and surveillance, infrastructure inspection and emergency response, among others. Although state-of-the-art deep learning-based object detectors (e.g., Faster R-CNN [27], Cascade R-CNN [3] and RetinaNet [20]) can be directly applied to UAV datasets (e.g., VisDrone [35] and UAVDT [9]), previous studies [17, 31, 15] reveal that these detectors do not perform well on drone-captured scenes due to two main challenges: (1) targets appear very small in high-resolution UAV images; (2) targets have a nonuniform spatial distribution in images. To address these challenges, a great amount of effort has been witnessed on augmenting the input images, such as developing effective image cropping strategies [17, 31, 15, 25]. For example, DMNet [17] generates a density map for each image and utilizes it to crop the original image to patches based on the object density. The goal is to make objects evenly distributed in each cropped patch. The detection results are fused from the global image and cropped patches in the test phase.

Although cropping methods alleviate the issue caused by uneven spatial distribution of the targets, *they overlook another critical problem of UAV images*: object sample imbalance among categories. As shown in Table 1, a few classes

*Equal contribution

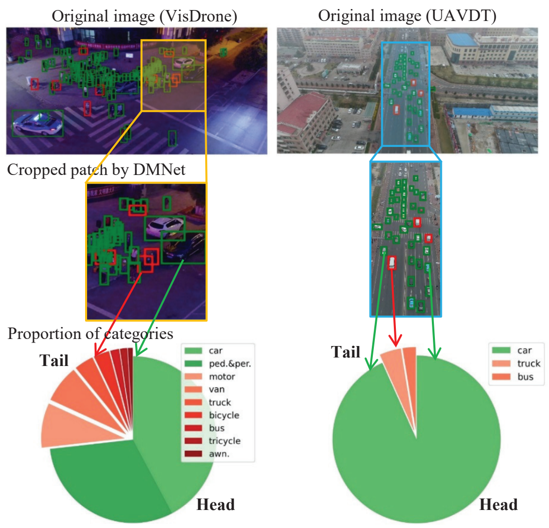

Figure 1. UAV datasets (e.g., VisDrone [35] and UAVDT [9]) manifest the long-tail distribution phenomenon. The green boxes in these images denote the head-class objects and the red boxes denote the tail-class objects. The patches in the middle are generated by the DMNet [17] cropping method. It can be observed that class imbalance exists in both original images and cropped patches. In other words, cropping methods only solve the problem of spatial nonuniform distribution of targets, but does not take into account the issue of class imbalance.

	ped	person	bicycle	car	van	truck	tricycle	awn.	bus	motor
Number of targets	79337	27059	10480	144867	24956	12875	4812	3246	5926	29647
Proportion (%)	23.1	7.9	3.1	42.2	7.2	3.8	1.4	0.9	0.7	8.6
FRCNN (average precision)	21.4	15.6	6.7	51.7	29.5	19.0	13.1	7.7	31.4	20.7

Table 1. Imbalanced distributions of objects/targets in VisDrone [35] training set. Objects marked in red color belong to the tail classes while those in blue belong to the head classes. **Note**: ped (pedestrian) and $person$ are semantically the same, therefore we put $person$ in head classes as ped. In this table, we also show the average precision of each individual class based on Faster R-CNN [27] (FRCNN for short). It is evident that there is a big performance gap between head classes and tail classes (e.g., car vs. awn. = 51.7 vs. 7.7).

in VisDrone [35] such as car, $ped.$ and $person$ account for more than 70% of all the targets (these classes are known as head classes), while other classes such as $tricycle$ and $awn.$ have only a small number of samples (i.e., tail classes). This is referred to as the long-tail distribution. Based on the detection results of Faster R-CNN [27] in Table 1, we can easily notice the performance gap between head classes and tail classes (e.g., car vs. awn. = 51.7 vs. 7.7 in terms of average precision). Fig. 1 presents a few concrete examples to reveal the imbalanced class distribution in UAV images. The head-class objects (denoted by green boxes) dominate the scene. *Even with the state-of-the-art cropping approach (DMNet [17]), the resulting patches still exhibit imbalanced class distribution.*

Tackling the long-tail distribution problem is important and has been a hot research topic in general object detection. One common approach is to repeatedly sample targets of tail classes [12] or discard some targets of head classes on purpose [8]. However, these straightforward methods suffer distortion of the original distribution which impairs the representation learning [34, 32]. Therefore another line of research [34, 32] aims to balance the class distribution by using different input distributions for training representation and classifier respectively in two phases (in the context of object detection, representation refers to feature extraction network and classifier refers to box head). These methods effectively reduce the performance drop caused by long-tail distribution on natural scene datasets (e.g., LVIS [11]).

However, there are unique challenges that make the long-tail problem more difficult on UAV datasets than on general object detection datasets. Several state-of-the-art methods for long-tail visual recognition [28, 32] sample a balanced set of targets in a batch based on the assumption that they can use a relatively large batch-size. However, this assumption may not be warranted on UAV datasets because the images are high-resolution (e.g., VisDrone dataset has many images over the size of 2000×1500). Due to the memory constraint, the typical batch-size is set to 1 or 2 for model training on these UAV datasets. Moreover, there are often hundreds of targets from head classes in one image (see the UAV images in Fig. 1). As a result, image-level repeat sampling [28] is not an effective solution on UAV datasets.

In light of these challenges, we propose a novel Dual Sampler and Head Network (DSHNet) to address long-tail distribution in UAV datasets for object detection. DSHNet consists of two key components including Class-Biased Samplers (CBS) and Bilateral Box Heads (BBH). Different from image-level re-sampling, CBS use the biased-sampling strategy to sample tail-class and head-class proposals separately with two samplers instead of a (default) single sampler in existing object detectors. BBH then separate the sampled tail-class and head-class proposals and process them with two box heads in the training phase to compute their losses respectively. In the test phase, each head of BBH only predicts results of the corresponding (tail or head) classes.

Our main contributions can be summarized as follows:

- We unveil the long-tail distribution problem in UAV images, which greatly hinders the performance of object detection. Previous works [31, 17, 33] mainly focus on resolving the nonuniform spatial distribution by cropping the original images into small patches. However, the cropped images still exhibit long-tail distribution.
- We analyze the unique challenges of object detection in UAV images and uncover that the solutions to long-tail distribution in natural images are not trivially applicable to UAV images. In light of this, we propose a novel Dual Sampler and Head Network (DSHNet) to handle head classes and tail classes separately.
- DSHNet substantially outperforms the baseline models and generic long-tail solutions on various object detectors and network backbones. When coupled with image cropping methods (e.g., DMNet [17]), DSHNet further improves the detection performance, resulting in new state-of-the-art results on VisDrone and UAVDT benchmarks.

2. Related work

General object detection. Deep learning-based object detection frameworks are divided into anchor-free and anchor-based ones. Anchor-free approaches [10, 16] focus on detecting objects by locating and regressing key points. Anchor-based methods can be further grouped into two-stage [27, 3, 13, 7] and one-stage [20, 22, 26] detectors. The two-stage methods separate the training phase of detection into two steps: (1) using feature extraction network and anchor generator to produce candidate regions (i.e., region proposals); (2) utilizing box regression head to refine the results of step (1) and compute the loss. After the first step, the network usually adopts a sampler to sample some objects instead of training all of them to keep a balance between background and foreground proposals and to reduce computation. In one-stage methods, detectors directly regress the location and bounding box from anchors without candidate regions.

Object detection in UAV images. Compared with natural images, object detection in UAV images is more challenging. The performance of the generic object detectors is degenerated due to the spatial nonuniform distribution of targets and small target size. To tackle this issue, many approaches [17, 31, 15, 25] generate a set of sub-images based on cropping methods. The general process of cropping-based methods is first using a proposal sub-net to analyze spatial information of objects and crop an image into small patches, and then training existing detection models with these patches. In the test phase, the final detection is obtained by fusing the detection results of local patches and global images with certain rules (e.g., non-maximum suppression (NMS)).

Another fundamental challenge lies in the imbalanced class distribution in UAV datasets, which leads to poor performance on tail classes as shown in Table 1. Only a few works [5, 33] have touched upon this problem, yet they didn't address it from the perspective of solving long-tail distribution. For example, Zhang *et al.* [33] simply separate all the classes into two sub-categories and train two networks individually, which harms the generalization of representation and classifier due to discarding too many samples in training each network.

Long-tail object detection. Most methods for long-tail object detection [28, 18, 29] come from long-tail classification [32, 23, 34], because the idea of dealing with the imbalanced class distribution is consistent. The following two approaches are considered to be the most effective ones:

- **Re-sampling** [1, 2]: The main idea of re-sampling is to over-sample tail classes or under-sample head classes to balance the data distribution, thereby improving the chance of tail classes being trained. But sometimes, with re-sampling, duplicated samples of tail classes might lead to over-fitting, while discarding samples of head classes would impair the generalization ability of network.
- **Re-weighting** [6]: Re-weighting methods assign large weights for training samples of tail classes or hard instances in loss functions. However, re-weighting is not able to handle large-scale datasets since it can cause optimization difficulty [24], leading to poor performance.

Beyond these methods, the authors of [32, 34] have shown that re-balanced input distribution improves classifier learning but harms feature learning. Therefore, many methods [34, 32, 28] adopt the two-phase training paradigm: first train on the original data distribution normally; then fine-tune the classifier on a balanced data distribution with fixed representation. The current solutions to long-tail distribution are generally based on this paradigm.

But if we simply regard the long-tail issue in detection as the one in classification, then at least, we need to assume that in each batch, the numbers of targets of different classes are roughly the same in each image. Unfortunately, this assumption is difficult to hold for object detection on UAV datasets since there is a huge gap in the numbers of targets of different classes as demonstrated in Fig. 1.

3. Methodology

3.1. Long-tailed object detection in UAV images

We observe that existing re-sampling approaches for long-tailed object detection are based on the input of the model, namely image-level. To avoid over-fitting, [34, 28, 32] propose to create a balanced input batch instead of simply repeating tail-class sampling or discarding head-class samples in each training iteration. These methods seem to alleviate the long-tail issue in general object detection, but their effectiveness depends on two requirements: (1) these methods require a relatively large batch-size so that there are enough targets of different categories to keep a balanced input batch; (2) they assume that each image contains a similar number of targets of different categories, otherwise the methods will degenerate to the simple repeat sampling or under-sampling which results in over-fitting or wasting training data. However, these two requirements are hard to meet in UAV object detection.

To be more specific, SimCal [28], a representative re-sampling approach, proposes to sample around 20% classes in each batch (i.e., 16 out of 80 classes for MS COCO [21]), so the batch-size is 16 where one image for each class. However, UAV images are usually in high resolution (e.g., a common resolution of 2000×1500 for the VisDrone dataset). Therefore, only 1 or 2 images can be fit into one batch. Considering that the loss is computed by averaging over all sampled classes in each batch, a relatively large number of classes should be included in each batch. Even if the first requirement can be met by sufficient computing resources, the second requirement cannot hold in most UAV images. The long-tail issue in UAV images does not only occur among images (i.e., inter-image) but also within images (i.e., intra-image), because there are frequently too many objects belonging to the head classes in one image (see Fig. 1 for an example). It is not because there are more head-class images than tail-class images as in general object detection scenarios. In this case, even if these methods have a balanced class distribution in terms of images, the objects of different classes are still highly imbalanced.

The Multi-Model Fusion (MMF) method in [33] is a spe-

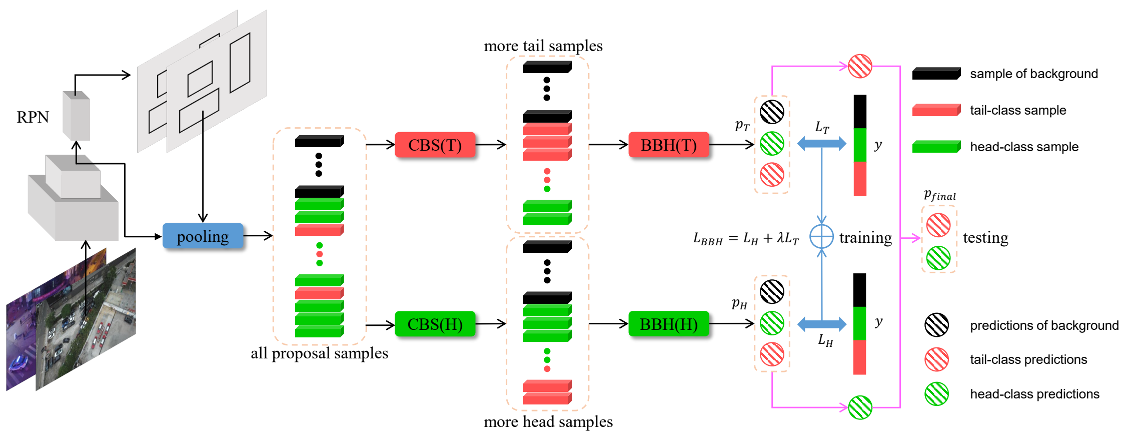

Figure 2. The proposed DSHNet pipeline based on Faster R-CNN. RPN denotes region proposal network. CBS(*) denote tail(T)-biased and head(H)-biased samplers, and BBH(*) represent their corresponding box heads. The part after BBH shows that each box in BBH computes loss of all classes in the training phase, while only gives results of the corresponding (tail or head) classes in the inference phase.

cific solution to imbalanced class distribution in UAV images. It separates all the classes into two groups by the amount of targets in the dataset. Then two detectors are trained for the two groups of classes separately. In the test phase, MMF fuses the results from two detectors. MMF discards a lot of useful data in training each model, which impairs the model representation capability. For example, when the first model learns information about *car*, because the corresponding ground truth does not include *van*, the representation cannot learn the difference between the two classes well. So the performance only improves slightly over the base model (see Table 2).

3.2. Overview of DSHNet

The proposed DSHNet is a plug-and-play method for all anchor-based one-stage or two-stage detectors. It has two key components: Class-Biased Samplers (CBS) and Bilateral Box Heads (BBH). The overall pipeline based on Faster R-CNN is depicted in Fig. 2. Different from the random sampler in Faster R-CNN, DSHNet has two biased samplers. CBS(H) samples head classes in priority while CBS(T) samples tail classes preferentially. After CBS, the two groups of biased samples are fed into BBH(H) and BBH(T) correspondingly. During training, BBH compute the loss over all classes which is the same as the box head in Faster R-CNN. While during inference, BBH only predict the results for the corresponding (head or tail) classes and the predictions are fused to obtain the final results. The feature extraction backbone and region proposal network (RPN) are the same as Faster R-CNN.

3.3. Class-biased samplers

To handle intra-image object imbalance in UAV images, we propose the Class-Biased Samplers (CBS) to perform re-sampling on the object-level when generating object proposals. The CBS module consists of two biased proposal samplers. The first one is CBS(T) which samples tail classes in priority. This means we first collect proposals which belong to the tail classes. If the number of proposals is insufficient after collecting all the tail-class proposals, CBS(T) continues to collect other classes. For example, in Faster R-CNN, the sampler will randomly sample 512 proposals, where 25% of them are positive samples and 75% of them are negative samples. CBS(T) will sample tail classes first. After that if the number of proposals is less than 128, the sampler will continue to sample targets of head classes. The second biased sampler is CBS(H), which works in the same scheme as CBS(T) by changing tail classes to head classes. Please refer to Alg. 1 for more details. It should be noted that the performance improvement of CBS is not due to the increased number of samples. As shown in the ablation study in Sec.4.4, increasing the number of samples in the random sampler can not benefit the performance.

3.4. Bilateral box heads

Since the CBS module generates two groups of biased samples, we propose the Bilateral Box Heads (BBH) to process the tail-biased and head-biased proposals respectively. As demonstrated in [34, 32], the backbone network (for feature extraction) prefers the original data distribution to learn well generalized representations. While the classifier tends to perform better on the biased classes in the data. By introducing BBH, we use two classifiers to cope with head classes and tail classes respectively. This allows each classifier to perform better on corresponding classes. Besides, CBS and BBH do not change the input distribution of the backbone network, which helps the backbone network to learn better generalized representations compared to existing image-level methods [34, 28, 29].

Algorithm 1: Class-biased samplers

```
Input: samples of tail classes $S_t$, samples of head classes $S_h$,
       samples of background $S_b$, number of required samples
       $N_s$, positive fraction $\alpha$
Output: Results of CBS(T) $R_t$, results of CBS(H) $R_h$
$Random(set, num)$ means if $num < length(set)$ randomly
  return $num$ elements from $set$; else return all the $set$ ;
begin
    ▷initialization
    $R_t \leftarrow \{\}, R_h \leftarrow \{\}$;
    ▷compute numbers of required positive and negative samples
    $N_p = N_s \times \alpha, N_n = N_s - N_p$;
    ▷first add samples of background and tail classes to the
      results of CBS(T)
    $R_t \leftarrow R_t \cup Random(S_b, N_n)$ ;
    $R_t \leftarrow R_t \cup Random(S_t, N_p)$ ;
    ▷then add samples of head classes to the results of CBS(T) if
      needed
    if $length(S_t) < N_p$ then
        $R_t \leftarrow R_t \cup Random(S_h, N_p - length(S_t))$;
    end
    ▷repeat steps above for CBS(H)
    $R_h \leftarrow R_h \cup Random(S_b, N_n)$ ;
    $R_h \leftarrow R_h \cup Random(S_h, N_p)$ ;
    if $length(S_h) < N_p$ then
        $R_h \leftarrow R_h \cup Random(S_t, N_p - length(S_h))$;
    end
end
```

BBH can also re-weight the losses of head classes and tail classes. First, in each head of BBH, there are more samples of corresponding (head or tail) classes than those in the original box head, so the weight of corresponding classes is improved. Second, we can manually adjust the ratio between the losses of two heads. The final loss function of BBH is

$$L_{BBH} = L_H(p_H, y) + \lambda L_T(p_T, y), \tag{1}$$

where L_T and L_H are the loss functions of BBH(T) and BBH(H), respectively. p_T and p_H denote the predictions of BBH(T) and BBH(H) respectively, including box regression and class score. y represents the label of bounding box and class. λ is a balance coefficient.

During training, both BBH(T) and BBH(H) compute losses on all classes since including other classes is beneficial to the generalization of the classifier. During inference, BBH(T) and BBH(H) only make predictions for its corresponding classes (see Fig. 2), and the two predictions are aggregated as the final results.

3.5. Plug-and-play on base models

Fig. 2 shows how to implement DSHNet on two-stage detectors. DSHNet can be easily applied to other mainstream detection pipelines including the one-stage and cascaded detectors without additional bells and whistles, as described in the following.

- **One-stage detectors**: One-stage detectors directly perform regression and classification for every anchor in the image instead of generating ROIs (region of proposal, like Faster R-CNN [27]). Take RetinaNet [20] for example, the final Retina head computes loss of all the targets. In DSHNet, we use CBS to assign label weights as 1 to sampled targets, so that other targets with label weights as 0 will not be considered in computing the loss.
- **Detectors with cascade architecture**: Cascade architecture means in the second stage of detection, networks have multiple (usually 3) box heads to process ROIs. Take Cascade R-CNN for example, in the second stage, there are 3 samplers for each box head. In DSHNet, we double all of them which means we have 6 samplers and 6 box heads (3 for tail classes and 3 for head classes). In the training phase, CBS and BBH work as same as them in Faster R-CNN. In the inference phase, class scores are computed by averaging over the 3 heads of BBH while box regressions are the results of the latest head of BBH of corresponding classes.

4. Experiments

4.1. Datasets

To validate the effectiveness of DSHNet, we conduct extensive experiments on two popular benchmarks for object detection in UAV images: VisDrone [35] and UAVDT [9].

- **VisDrone**: The dataset consists of 10,209 images (6,471 for training, 548 for validation and 3,190 for testing) with rich annotations on ten categories of objects. The image scale of the dataset is about 2,000 × 1,500 pixels. Since the evaluation server is closed now, we cannot test our method on the test set. Therefore, the validation set is used to evaluate our method, which is a setting adopted by previous methods as well.
- **UAVDT**: The UAVDT [9] dataset contains 23,258 images for training and 15,069 images for testing. The resolution of the image is about 1,080 × 540 pixels. The dataset is acquired with an UAV platform at a number of locations in urban areas. The categories of the annotated objects are car, bus, and $truck$.

4.2. Implementation details

We implement DSHNet based on the MMdetection [4] toolbox. Faster R-CNN (FRCNN) [27], RetinaNet (Retina) [20] and Cascade R-CNN (CRCNN) [3] with Feature Pyramid Network (FPN) [19] which are representatives of two-stage detectors, one-stage detectors and detectors with cascade architecture, are adopted as the baseline detection networks. The parameters of CBS are basically the same as the original random sampler (RS, randomly sample the corresponding number of positives and negatives) in the implementation of Faster R-CNN in MMdetection:

Method	backbone	AP	AP_{50}	AP_{75}	ped.	person	bicycle	car	van	truck	tricycle	awn.	bus	motor
Comparison with base models														
RetinaNet+RS	R50	13.9	27.7	12,7	13.0	7.9	1.4	45.5	19.9	11.5	6.3	4.2	17.8	11.8
FRCNN+RS	R50	21.7	39.8	21.0	21.4	15.6	6.7	51.7	29.5	19.0	13.1	7.7	31.4	20.7
FRCNN+RS	R101	21.8	40.2	20.9	20.9	14.8	7.3	51.0	29.7	19.5	14.0	8.8	30.5	21.2
FRCNN+RS	X101	22.4	41.0	21.8	21.3	15.5	7.9	52.0	29.5	20.5	14.7	8.9	32.1	21.6
CRCNN+RS	R50	23.2	40.7	23.1	22.2	14.8	7.6	54.6	31.5	21.6	14.8	8.6	34.9	21.4
RetinaNet+DSHNet	R50	16.1	30.2	15.5	14.1	8.9	1.3	48.2	24.8	14.2	8.8	6.0	21.6	13.1
FRCNN+DSHNet	R50	24.6	44.4	24.1	22.5	16.5	10.1	52.8	32.6	22.1	17.5	8.8	39.5	23.7
FRCNN+DSHNet	R101	24.4	44.3	23.8	21.7	16.0	10.1	52.2	31.6	22.7	17.1	9.5	38.6	24.0
FRCNN+DSHNet	X101	25.8	**46.8**	25.2	**23.3**	**16.7**	**11.4**	53.7	33.1	23.8	**19.5**	**11.1**	40.0	**25.5**
CRCNN+DSHNet	R50	**26.2**	45.0	**26.3**	23.2	16.1	11.2	**55.5**	**33.5**	**25.2**	19.1	10.0	**43.0**	25.1
Comparison with solutions to long-tail problems														
FRCNN+RS+MMF [33]	R50	22.6	41.7	21.6	21.6	15.3	9.6	51.5	28.5	20.4	15.9	7.5	33.7	21.6
FRCNN+SimCal [28]	R50	20.0	35.8	19.6	18.7	13.8	5.7	51.0	28.4	16.4	13.6	5.9	27.0	19.4
FRCNN+RS+BGS [18]	R50	23.0	43.0	22.0	21.8	16.0	8.1	51.8	31.1	19.8	15.0	8.4	36.1	21.5
FRCNN+DSHNet	R50	**24.6**	**44.4**	**24.1**	**22.5**	**16.5**	**10.1**	**52.8**	**32.6**	**22.1**	**17.5**	**8.8**	**39.5**	**23.7**
Based on the SOTA cropping method														
DMNet (FRCNN+RS) [17]	R50	28.1*	48.5	28.1	28.1	19.7	13.3	**57.3**	36.1	24.8	20.1	12.0	42.9	26.4
DMNet [17] cropping+DSHNet	R50	**30.3**	**51.8**	**30.9**	**28.5**	**20.4**	**15.9**	56.8	**37.9**	**30.1**	**22.6**	**14.0**	**47.1**	**29.2**

* Note: we reproduce the DMNet [17] result using authors' implementation and default settings since the original DMNet paper didn't report the class-wise AP. Our reproduced AP is 28.1, which is only 0.1 less than the result (28.2) in the original paper.

Table 2. The detection performance on VisDrone validation set. RS is short for random sampler.

the total number of samples is 512, and 25% of the samples are positive and 75% are negative. In VisDrone [35], car, $ped.$ (pedestrian) and $people$ are considered as head classes (noted that the boundary between $ped.$ and $people$ in VisDrone is ambiguous, so we group them together), while other object categories are regraded as tail classes. In UAVDT [9], car is the head class and $truck$ and bus are the tail classes. The BBH is the same as the original box head in Faster R-CNN which has 2 shared fully connected layers and 1 fully connected layer for predicting class scores and box regressions, respectively.

Training phase. The input size of the detector is 1,000 $\times$ 600 pixels on both VisDrone [35] and UAVDT [9] datasets. The batch-size is set to 2 (i.e., 2 images) on a single NVIDIA 1080Ti GPU with 11GB memory. On VisDrone, we set the base learning rate to 0.002 and the training epochs to 18. After the 8th, 12nd and 16th epoch, the learning rate decreases by a factor of 10. On UAVDT, we set the training epochs to 4 and fix the learning rate at 0.002.

Test phase. The input size of detector is the same as that in the training phase. The maximum detection number is set to 500 by following the settings of VisDrone [35]. Following the evaluation protocol on MS COCO [21], we use AP, AP_{50}, and AP_{75} as metrics to measure the precision. AP_{50} and AP_{75} are computed at the single IoU threshold 0.5 and 0.75 over all categories. In addition, we report the average precision of each object category.

4.3. Experimental results

Comparison with base models. We implement 3 base models: Faster R-CNN (FRCNN) [27], RetinaNet (Retina) [20] and Cascade R-CNN (CRCNN) [3] according to the same settings of DSHNet. For the backbone network, we choose ResNet50 (R50), ResNet101 (R101) [14] and ResNeXt101 (X101) [30], and use the default parameters (including the feature pyramid network (FPN) [19]) in MMdetection [4].

Table 2 reports the overall AP and APs of all the 10 classes of VisDrone. DSHNet achieves consistent improvements on all the base models. On the most representative two-stage detector, Faster R-CNN, we conduct experiments using 3 different backbone networks, and the performance increases the most on ResNeXt101 (22.4 vs. 25.8). On the state-of-the-art Cascade R-CNN (CRCNN) model, our method also gains a considerable 3.0 overall AP (23.2 vs. 26.2). Moreover, in almost all the cases, AP of each class is improved compared with the counterpart in the base models. This demonstrates that both head and tail classes can benefit from DSHNet. For tail classes in particular, the detection results are improved significantly. For example, our DSHNet boosts the APs of tail classes $tricycle$ and bus by about 29% (14.8 vs. 19.1) and 23.2% (34.9 vs. 43.0), respectively, based on the CRCNN (R50). We also present a visual comparison of the detection results on VisDrone dataset (Fig. 3). In this example, the detection precision of $motor$, one of the tail classes, is obviously improved by DSHNet as compared to the base model.

On UAVDT, we conduct experiments based on RetinaNet [20] and Faster R-CNN [27]. Similar to the results in VisDrone, DSHNet achieves remarkable improvements over baseline models, and the precision of both head classes and tail classes are largely improved. Specifically, the tail class bus is dramatically improved from 3.4 to 14.7 on FRCNN-R50 as shown in Table 3. This clearly validates

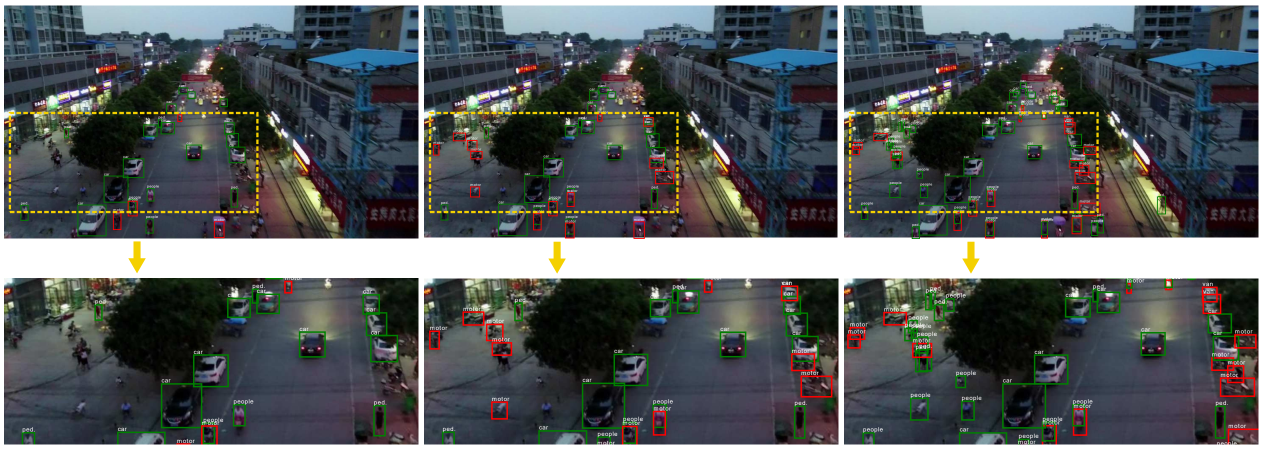

Figure 3. Visualization of detection results on VisDrone. From the first to third columns are base model (Faster R-CNN (ResNet50)), DSHNet on the base model, and ground truth (red boxes for tail classes and green boxes for head classes). The detection precision of tail-class *motor* is greatly improved by DSHNet. (Best viewed on screen with zoom.)

Method	backbone	AP	AP_{50}	AP_{75}	car	truck	bus
FRCNN+RS	R50	11.0	19.9	11.2	28.3	1.4	3.4
FRCNN+RS	R101	12.5	22.3	12.9	27.9	1.9	7.8
Retina+RS	R50	14.5	28.5	12.9	30.0	3.7	9.7
FRCNN+DSHNet	R50	16.0	28.5	16.3	29.4	3.9	14.7
FRCNN+DSHNet	R101	17.0	30.0	18.1	29.8	3.4	**17.9**
Retina+DSHNet	R50	**17.8**	**30.4**	**19.7**	**32.1**	**4.2**	17.0

Table 3. The detection performance on UAVDT validation set.

the effectiveness of our proposed DSHNet.

Comparison with existing solutions tackling long-tail problems. We compare our DSHNet with state-of-the-art methods for long-tail visual recognition. In MMF [33] and BGS [18], tail and head classes are defined according to the same rules as DSHNet (see Sec. 4.2 for detail). In SimCal [28], we choose class-agnostic method which has better performance on overall AP. We assign 2 classes per batch for training in order to keep the sample ratio unchanged (2:10 on VisDrone [35] vs. 16:80 on MS COCO [21]).

We first compare our method with MMF [33], which also addresses the imbalanced class distribution in VisDrone dataset. The main idea of MMF is to divide the 10 classes to 2 sub-categories and train and test the model separately for those two sub-categories.The feature network of each model in MMF only gets a part of all the classes as input, instead of getting the original distributed data like DSHNet, which harms the generalization of representation. Therefore, this method only slightly improves the base model from the results in Table 2 (21.7 vs. 22.6). And the overall AP is 2 points worse than our DSHNet (22.6 vs. 24.6).

As discussed in Sec. 2, most existing methods tackling the long-tail problems on natural datasets do not work well on UAV datasets. Here, we choose two recent methods representing the re-sampling and re-weighting strategies, respectively. SimCal [28] is improved from image-level resampling which also uses class-aware sampler. However, SimCal [28] needs to sample enough categories and collect similar numbers of different targets to ensure a balanced distribution in a batch. SimCal [28] is not trivially applicable on UAV datasets, and the performance is even inferior to that of the base model (20.0 vs. 21.7).

BGS [18] is the representative re-weighting method which introduces the balanced group softmax to group corresponding classes together when computing loss. BGS is currently the best re-weighting solution to long-tail issue. However, BGS only achieve re-weighting in loss function while does not change the distribution of box head, which proved to effectively improve classifier learning [28, 32, 34]. Therefore, the imbalanced distribution of input cannot be re-balanced as good as DSHNet. Despite its better performance than the base model, BGS is still worse than DSHNet (23.0 vs. 24.6).

How DSHNet performs with image patches? As shown in Fig. 1, image cropping methods do not solve the long-tail distribution problem. Since image cropping can be considered as a data pre-processing procedure, our DSHNet can be equally applied to image patches, which could further improve the performance of cropping-based approaches. To evaluate DSHNet on image patches, we employ the state-of-the-art DMNet [17] to generate cropped patches and train DSHNet on them. The results in Table 2 (last two rows) show that DSHNet also works on image patches and boosts the overall AP by 2.2 points (30.3 vs. 28.1).

4.4. Ablation study

To validate the contributions of CBS and BBH to the improvement of detection performance, we carry out ablation experiments on VisDrone [35] dataset with Faster R-CNN [27] and ResNet50 [14].

Method	backbone	AP	AP_{50}	AP_{75}	ped.	person	bicycle	car	van	truck	tricycle	awn.	bus	motor
FRCNN+RS	R50	21.7	39.8	21.0	21.4	15.6	6.7	51.7	29.5	19.0	13.1	7.7	31.4	20.7
FRCNN+RS-DBL	R50	21.6	40.1	20.3	20.9	15.4	6.6	52.1	27.4	18.9	13.5	8.0	32.2	21.5
FRCNN+RS-DBL+BBH	R50	22.1	41.1	21.2	21.1	15.0	7.7	51.6	29.1	18.9	15.2	8.3	33.1	20.9
FRCNN+CES+BBH	R50	24.1	43.7	23.7	21.9	15.8	9.4	52.1	30.8	22.0	**17.7**	**9.7**	39.5	22.5
FRCNN+CBS+BBH	R50	**24.6**	**44.4**	**24.1**	**22.5**	**16.5**	**10.1**	**52.8**	**32.6**	**22.1**	17.5	8.8	**39.5**	**23.7**

Table 4. The ablation study of CBS on VisDrone dataset.

Method	backbone	AP	AP_{50}	AP_{75}	ped.	person	bicycle	car	van	truck	tricycle	awn.	bus	motor
FRCNN+RS	R50	21.7	39.8	21.0	21.4	15.6	6.7	51.7	29.5	19.0	13.1	7.7	31.4	20.7
FRCNN+CBS	R50	21.7	40.2	20.8	20.5	14.8	6.9	51.6	29.0	19.1	13.9	8.3	31.6	20.7
FRCNN+CBS+BBH-ALL	R50	22.6	40.0	22.7	16.2	14.5	**10.1**	47.8	32.0	19.6	17.1	**8.8**	37.2	23.4
FRCNN+CBS+BBH	R50	**24.6**	**44.4**	**24.1**	**22.5**	**16.5**	**10.1**	**52.8**	**32.6**	**22.1**	**17.5**	**8.8**	**39.5**	**23.7**

Table 5. The ablation study of BBH on VisDrone dataset.

Effect of CBS. In order to verify that the performance of the detector is not simply related to the number of samples generated by the sampler, we conduct two experiments. First, we double the number of samples for the random sampler in Faster R-CNN [27] (denoted by FRCNN+RS-DBL) in Table 4. The performance is on par with the base model (21.6 vs. 21.7), indicating that simply increasing the number of samples will not improve the performance when the sample size is adequate. Second, we remove the CBS module and use the random sampler to obtain two groups of samples as input to BBH (denoted by FRCNN+RS-DBL+BBH in Table 4). The performance is only slightly better than the base model (22.1 vs. 21.7), which verifies the crucial role of CBS in proposal re-sampling.

Another question we would like to answer is "which sampler is better, class-biased or class-exclusive?". Class-exclusive sampler (CES) means that we only sample assigned classes without other classes. More concretely, after CBS(T) samples all the tail-class samples, if the number of samples does not meet the requirement, it will continue to sample head-class proposals, but CES will not. The performance of CES is lower than CBS (24.1 vs. 24.6) which shows that if not affecting the main classes, adding other classes is beneficial to the generalization of the classifiers.

Effect of BBH. To show that BBH can enhance the discriminative power of the classifiers, we remove BBH and simply integrate the results of CBS as input to the original single box head (FRCNN+CBS in Table 5). The performance of the network without BBH drops considerably compared to that of the complete DSHNet (21.7 vs. 24.6).

In the test phase, BBH(T) or BBH(H) only makes predictions for tail or head classes before the results aggregation. To validate the advantage of this prediction strategy in BBH, we establish a comparison method by allowing both BBH(T) and BBH(H) to generate predictions for all classes (denoted by FRCNN+CBS+BBH-ALL in Table 5) and fusing the predictions by NMS. The separate prediction strategy improves the AP by 2 (22.6 vs. 24.6). It is because each head of BBH is trained with class-biased samples and by only considering the class predictions that they are good for (i.e., tail-class predictions from BBH(T) and head-class predictions from BBH(H)) would achieve better performance.

Effect of λ. The coefficient λ in the loss function of BBH (see Eq. 1) can be optimized for performance. As shown in Fig. 4, when $\lambda < 1$, the performance is relatively poor because the tail classes have less weight in the training. Between 2.0 and 4.0, the performance remains at the highest AP (24.6). Then the performance declines when $\lambda > 4.0$. In the experiments, we set $\lambda = 2.0$.

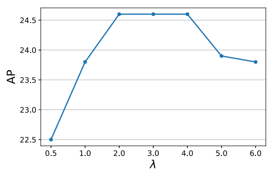

Figure 4. The AP over different settings of λ in the loss of BBH.

Inference speed. We report the inference speeds of base models and DSHNet based on the implementations of MMdetection [4] on one NVIDIA 1080Ti GPU (see Table 6). DSHNet is only slightly slower than base models.

Model	base		DSHNet	
	fps	AP	fps	AP
RetinaNet [20]	22.5	13.9	16.4	16.1
Faster R-CNN [27]	20.1	21.7	15.7	24.6
Cascade R-CNN [3]	14.9	23.2	10.8	26.2

Table 6. The inference speed (frames per second) and AP of base models and DSHNet with ResNet-50.

5. Conclusion

We proposed a novel Dual Sampler and Head detection Network (DSHNet) to solve the long-tail distribution problem in UAV datasets. The Class-Biased Samplers (CBS) are introduced to perform biased sampling on object proposals for tail and head classes respectively. The Bilateral Box Heads (BBH) use two classifiers to process the tail-biased and head-biased proposals separately. Moreover, BBH achieve loss re-weighting by computing the loss for head and tail classes respectively. Experiments on two UAV benchmarks show that our method significantly improves the base models and achieves new state-of-the-art results.

References

[1] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[2] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881, 2019.

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[5] Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. *arXiv preprint arXiv:2008.03673*, 2020.

[6] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019.

[7] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[8] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8. Citeseer, 2003.

[9] Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 370–386, 2018.

[10] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6569–6578, 2019.

[11] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5356–5364, 2019.

[12] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Sungeun Hong, Sungil Kang, and Donghyeon Cho. Patch-level augmentation for object detection in aerial images. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[16] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018.

[17] Changlin Li, Taojiannan Yang, Sijie Zhu, Chen Chen, and Shanyue Guan. Density map guided object detection in aerial images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 190–191, 2020.

[18] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10991–11000, 2020.

[19] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[22] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[23] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[25] F Ozge Unel, Burak O Ozkalayci, and Cevahir Cigla. The power of tiling for small object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[28] Tao Wang, Yu Li, Bingyi Kang, Junnan Li, Junhao Liew, Sheng Tang, Steven Hoi, and Jiashi Feng. The devil is in classification: A simple framework for long-tail instance segmentation. In *European conference on computer vision*, 2020.

[29] Tao Wang, Yu Li, Bingyi Kang, Junnan Li, Jun Hao Liew, Sheng Tang, Steven Hoi, and Jiashi Feng. Classification calibration for long-tail instance segmentation. *arXiv preprint arXiv:1910.13081*, 2019.

[30] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[31] Fan Yang, Heng Fan, Peng Chu, Erik Blasch, and Haibin Ling. Clustered object detection in aerial images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8311–8320, 2019.

[32] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. *arXiv preprint arXiv:2006.07529*, 2020.

[33] Xindi Zhang, Ebroul Izquierdo, and Krishna Chandramouli. Dense and small object detection in uav vision based on cascade network. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[34] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2020.

[35] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Qinghua Hu, and Haibin Ling. Vision meets drones: Past, present and future. *arXiv preprint arXiv:2001.06303*, 2020.