Object Detection in Aerial Images: A Case Study on Performance Improvement

Adnan Khan*
Department of Computer Vision
MBZUAI
Abu Dhabi, UAE
adnan.khan@mbzuai.ac.ae

Muhammad Uzair Khattak*
Department of Computer Vision
MBZUAI
Abu Dhabi, UAE
uzair.khattak@mbzuai.ac.ae

Khaled Dawoud*
Department of Computer Vision
MBZUAI
Abu Dhabi, UAE
khaled.dawoud@mbzuai.ac.ae

***Abstract*—Object Detection (OD) in aerial images has gained much attention due to its applications in search and rescue, town planning, agriculture yield prediction, and many other areas. A recently introduced large-scale aerial images dataset, iSAID, has enabled researchers to advance the OD tasks on satellite images. Unfortunately, the available OD pipelines and ready-to-train architectures are well-tailored and configured for the tasks dealing with natural images. In this work, we study that directly using the available object detectors, specifically the vanilla Faster RCNN with FPN, is sub-optimal for aerial OD. To effectively improve its performance, we adapt the Faster R-CNN architecture. We propose several modifications, including changes in architecture in different blocks of the detector, training & transfer learning strategies, loss formulations, and other pre & post-processing techniques. By adopting the proposed design changes on top of the vanilla Faster-RCNN, we push the model's performance and achieve an absolute gain of 4.44 AP over the vanilla Faster R-CNN on the iSAID validation set. Our code is publically available at https://github.com/muzairkhattak/OD-Satellite-iSAID.**

I. Introduction

Object Detection (OD), in computer vision, is the task of classifying and localizing the objects within the images and videos. Given an image, OD aims to localize objects by assigning bounding box locations, and then it classifies each localized object [1]. Due to its widespread applications, OD has gained much attention in recent years. A new and emerging task nowadays is detecting objects in aerial images, which has its unique challenges compared to natural images [2]. The availability of large-scale satellite imagery datasets such as DOTA [3] and iSAID [4] has enabled researchers to progress on the challenging task of aerial OD.

Before the realm of Deep Learning (DL), the task of OD was tackled with traditional machine learning techniques, e.g., histogram of gradients (HoG) [5], which involved the designing of the hand-crafted features. After the DL leap, not only has the feature extraction become automated in the OD pipeline, but it has also improved the results substantially, leading to its success in different real-world applications. Therefore, DL is the de-facto method for developing OD-based applications.

In this work, we apply one of the most widely used OD architectures; Faster R-CNN [6], for detecting objects in a challenging satellite images dataset, namely the iSAID dataset. Furthermore, we modify the baseline architecture to adapt it for satellite images, leading to better performance. Our main study encloses the following:

1) We explore recently introduced vision backbones for OD, including ConvNext and SWIN transformer
2) We examine the use of different loss functions for OD
3) We investigate different transfer learning and pre-training techniques for better OD results
4) We investigate appropriate data augmentations for iSAID dataset
5) We perform a detailed experimental study on improving other various blocks of Faster-RCNN detector including region proposal network (RPN) and Region of Interest (RoI) Pooling.

II. Related Work

Object detection (OD) in aerial images is a hotline of research due to its potential applications and has gained much attention in the recent past. OD in satellite images differs from natural images due to the humongous amount of small objects available in these images [7]. These objects can be of irregular shapes and have existing orientation variations as shown in Fig. 1. Many state-of-the-art OD methods trained for natural images cannot quickly adapt to satellite images' varying shapes, scales, and orientations. This section briefly mentions the methodologies used for the OD tasks in natural and aerial images.

On a higher level, the OD methods are classified into two main types: single-stage object detectors and two-stage object detectors. The two-stage object detection frameworks [6], [8], [9], apply region proposal techniques to get regions of interest, then extract the features of the regions and then predict and regress the categories and bounding boxes, respectively. Single-stage detectors [10]–[13] require only a single pass to the neural network and directly get the detection results. Both families of detectors are widely used nowadays, with single-stage being much faster in inference time compared to two-

*Equal contribution.

979-8-3503-9676-8/22/$31.00 © 2022 IEEE

stage detectors. On the other hand, two-stage detectors yield higher results on detection tasks.

Recent studies have shown improvement in OD tasks with satellite images. An architecture based on a convolutional neural network (CNN) is presented in [14] for automatic image classification and detection in aerial images. Another recent work [15] proposes a novel two-stage detection, D2Det, to address both precise localization and accurate classification to obtain superior performance on object detection in satellite images. The proposed model with a ResNet101 [16] and FPN backbone is evaluated on the UAVDT dataset [17]. In [18], an architecture based on Fast R-CNN and Faster R-CNN is used to detect vehicles in aerial images and validate their results on two publicly available datasets. In [19], a cluster proposal-based network alleviates the problems of minor objects in aerial images. In our work, we similarly conduct experiments as followed in literature to improve OD performance on satellite images and modernize a vanilla Faster R-CNN by proposing several modifications, which leads to improved performance over the vanilla baseline model.

III. Methodology

A. iSAID Dataset

Aerial image datasets are steadily prevailing nowadays, and efforts are made to tailor the datasets so that real-life challenges with satellite images are addressed. For example, increasing the number of categories and the number of instances in these datasets is a crucial task to perform better on these images. Most popular of these datasets are: DIOR [20], DOTA [3], xVIEW [21] and iSAID [4]. The dataset chosen for our study is iSAID due to its unique challenges and higher number of categories and instances.

iSAID dataset is the first benchmark in the instance segmentation task of computer vision using aerial images. In iSAID, there are 2,806 high-resolution images with 6,55,451 object instances. There are a total of 15 categories: ground track field, bridge, large vehicle, small vehicle, helicopter, ship, storage tank, baseball court, tennis court, basketball court, swimming pool, roundabout, soccer ball field, plane, and harbor. The images are of large resolutions, which is computationally expensive for DL models to process. Therefore, a preprocessing step is used to address the issue of high resolution, in which patch sizes of 800x800 are extracted from the original images resulting in 28,029 and 9,512 images for training and validation, respectively. A few samples from the iSAID dataset are shown in Fig.1.

B. Baseline Model: FasterRCNN

We chose our baseline as the Feature Pyramid Network (FPN) based Faster R-CNN, a two-stage OD architecture. As shown in Fig. 2, Faster R-CNN contains a backbone and a region proposal network (RPN) followed by a region-of-interest (ROI) pooling block and prediction heads for classification and bounding box regression. The backbone consists of a convolutional neural network (CNN) and encodes a high-resolution input image into low-dimensional semantically rich features.

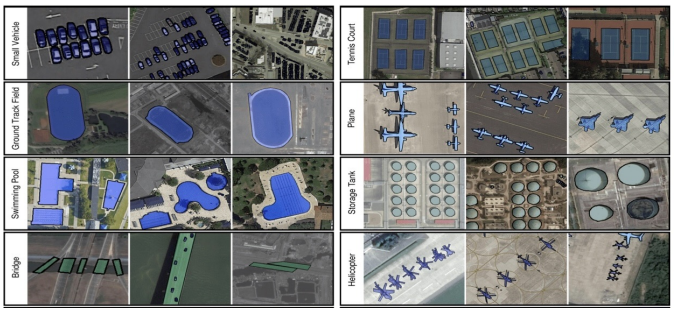

Fig. 1: Samples of annotated images in iSAID dataset [4]

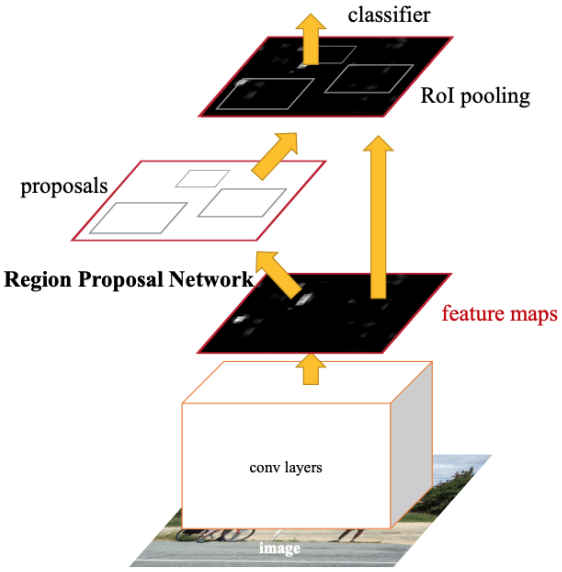

Fig. 2: Faster R-CNN Pipeline [6]. Image is encoded using CNN backbone and passed to RPN for object proposal generation. ROI pooling is used to extract region features which are passed to ROI head for classification and bounding box regression.

These features are passed to RPN for generating class-agnostic object proposals for the image. First, top N proposals (e.g., 1000, 2000) are selected from RPN. Then, ROI pooling is applied to extract the corresponding convolutional features, passed to output heads for object category classification and bounding box regression.

We choose a variant of Faster R-CNN that uses FPN. FPN utilizes bottom-up and top-down paths along with lateral connections in a convolutional neural network (CNN). It constructs semantically more robust features at multiple scales. To extract features at multiple levels, FPN-based Faster R-CNN uses the hierarchical property of convolution neural networks, which acts as a feature extractor. This feature extractor acts as the backbone of the detector. ResNet-101 [16] is used as the backbone in our baseline case. The multi-scale feature maps help the model to be trained on high-quality image data compared to the single-level feature map used in the detector.

Fig. 3 shows our overall baseline. The backbone network

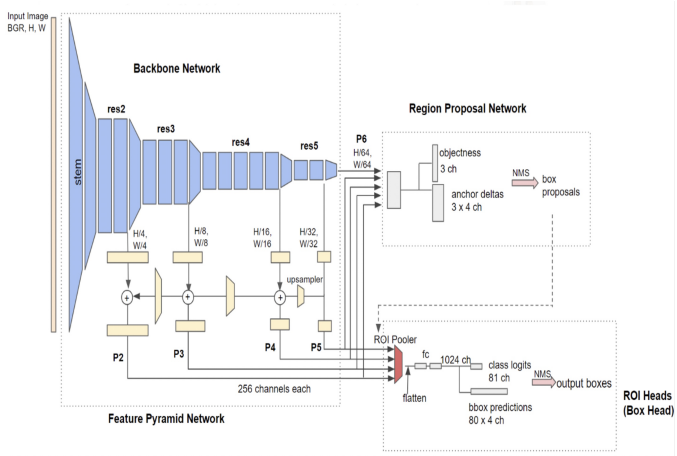

Fig. 3: Detailed architecture of Faster-RCNN-FPN. Best viewed when zoomed in.

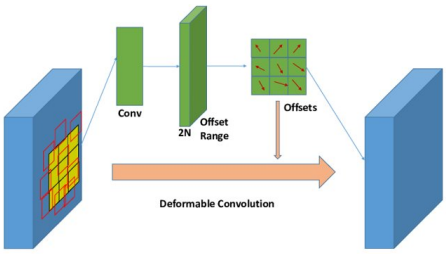

Fig. 4: Block diagram of a single deformable convolution layer

extracts feature maps from the input image at different scales. The output feature maps P2, P3, P4, P5, and P6 are generated with their corresponding scales having ratios of 1/4, 1/8, 1/16, 1/32 and 1/64, respectively. The non-FPN ('C4') architecture's output feature is only from the 1/16 scale. The backbone is next, followed by Region Proposal Network (RPN), which detects object regions from multi-scale features. Specifically, the anchor boxes strategy is used, which proposes candidate bounding boxes on top of the FPN feature maps, with various sizes and scales, along with its objectness scores. The baseline model uses anchor box sizes 32, 64, 128, 256, and 512. For each size, boxes are further replicated with three different aspect ratio scales, including 0.5, 1, and 2. RPN network detects the positive and negative samples from all anchor candidate samples. A total of 1000 box proposals, along with the confidence scores, are obtained by default. Following the RPN, the RoI pooling block generates the fixed-size features by cropping and warping the feature maps using the proposal boxes. Refined box locations and classification results are obtained in the second stage using fully connected layers. Finally, 100 boxes at maximum are filtered out using non-maximum suppression (NMS).

C. Modernization of Baseline Architecture and Training Strategy

With the chosen baseline FasterRCNN with R-101 imageNet-1k [22] pre-trained backbone, we propose several modifications in its different blocks to enhance the performance in the aerial object detection setting. Below we introduce each proposed modification in detail. In the experiment section, we study each modification separately and keep the modification in the following experiment only if the modification helps improve the performance; otherwise, the proposed changes are discarded in the next set of experiments.

1) Backbone Changes: Having a suitable backbone for rich feature extraction in the object detection pipeline is crucial, as the classification and bounding boxes regression estimates are done on the feature maps produced from the backbone. Different choices of the backbone are available in the literature. Here we explore backbones derived from competitive architectures, namely Deformable ResNet [23], SWIN transformer [24], and ConvNext [25].

Deformable-ResNet: Deformable convolution [23] can have a dynamic receptive field. As shown in Fig. 4, during the convolution operation, sampling of points via network filters is not restricted to a fixed square window; instead, it adaptively selects the feature points important in a neighborhood. For example, the nature of frames of the iSAID dataset, which has objects with high scale variation, as shown in Fig. 1. Deformable convolution layers are vital as they adaptively change the receptive field to cover the object, whether small or large. However, a critical issue in using deformable layers is that it poses a compute overhead as, firstly, it has to estimate the sampling points and then perform the convolution process. To reduce the overhead, we apply deformable convolution in only two out of five stages of standard ResNet architecture.

The deformable-ResNet model constitutes deformable convolutional layers instead of normal ones in its bottleneck layers. However, the rest of the architecture is the same as the original ResNet model. Specifically, we build upon ResNet-101 architecture and design its deformable variant, which we call Deform ResNet-101.

SWIN Transformer: Vision transformer (ViT) model based on shifted windows (SWIN) [24] has demonstrated competitive performance over other previously introduced vision transformers as well as CNN models. It uses SWIN blocks which contain shifted windows-attention layers. While restricting self-attention to a window using shifted windows over image patches brings more significant efficiency. Moreover, incorporating global interaction also allows the cross-attention between patches by using shifted windows scheme. In SWIN transformers, the patch merging technique proposes hierarchical architectures and is a common backbone choice.

ConvNext: Recently, another family of convolutional neural networks called ConvNext [25] has been introduced, which performs at par with the recent popular transformer-based image models, including SWIN. ConvNext is designed by building on top of the vanilla ResNet with modifications that are majorly inspired by the vision transformer architectures. Significant highlights of ConvNext include using long training schedules, strong data augmentations, and macro and micro designs. The macro design includes reshaping the architectural

aspects of ResNet to that of SWIN transformers, such as changing the stage compute ratio, using multi-layer perceptron (MLP) block instead of bottleneck block, and using aggressive down-sampling block in the stem layer. The micro designs include using the GELU activation functions instead of RELU, layer norms instead of the batch norm, etc. With the proposed model, ConvNext can achieve the SWIN transformer's performance in a scalable manner on various computer vision tasks, including classification and detection.

In this work, we explore the performance of these different choices to be used as a backbone for the task of aerial object detection.

2) Loss functions: Loss functions put the foundation of training on which the entire model is optimized. In the object detection pipeline, loss functions are formulated for classification and regression. We use different loss functions in our experiments which we briefly describe below.

Focal Loss for classification: Focal loss [26] is an extension over the normal cross-entropy loss, mainly designed to solve the class imbalance problem. Generally, positive examples (foreground) are fewer in numbers than negative examples (background). Therefore, standard cross-entropy loss demands a high confidence score for classifying positive examples. Due to the low number of positive examples, the network focuses more on understanding/differentiating the negative samples than positive ones. As a result, cumulative negative sample loss overwhelms the positive sample loss, which leads to degenerated model. Focal loss turns the model's attention to hard examples by increasing the loss for hard examples (which are mainly misclassified by the model) by assigning more weight to such cases. A comparison between focal loss and cross-entropy is given in Fig. 5.

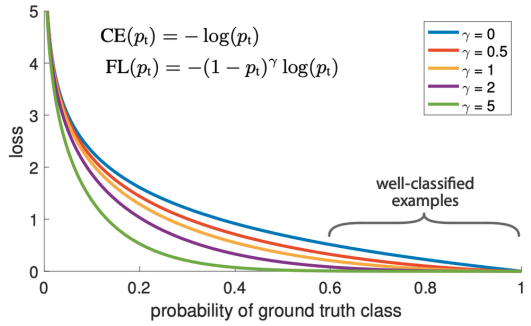

Fig. 5: Focal loss adds a factor of $(1-p_t)^\gamma$ to the standard cross-entropy loss. By setting $\gamma > 0$, the relative loss for well classified examples are reduced.

Federated Loss for classification: Federated loss [27], is a recently introduced loss formulation for class imbalanced dataset. As for an imbalanced dataset, some categories have significantly fewer annotations, while some are densely annotated. For a category having less annotation, the normal cross-entropy loss will pay more attention to the densely annotated classes (which the network is seeing more). However, it will not model balanced class learning attributes. Federated loss uses only a subset of the total number of categories based on the square root frequency of each class in the training set. During the loss calculation, all positive classes and the subset from the negative classes are used. For our experiments on the iSAID dataset, we set the negative class sample count (subset length) equal to 10.

GIoU Loss for Regression: Regression losses such as l1 and l2 are used for the model to learn to regress the bounding boxes accurately. They are evaluated based on the Intersection over Union (IoU) metric. However, the IoU formulation must better differentiate between a very bad box prediction and a relatively less bad prediction. Thus there is no strong correlation between minimizing these losses and improving the IoU metric. Generalized IoU [28] loss is formulated to consider cases where the IoU metric fails. Mathematically,

$$GIoU = \frac{|A \cap B|}{|A \cup B|} - \frac{|C/(A \cup B)|}{|C|} = IoU - \frac{|C/(A \cup B)|}{|C|} \quad (1)$$

Where A and B are the prediction and ground truth bounding boxes. C is the smallest enclosed area, which covers both A and B.

D. RoI Align Pooler Resolution

To provide the detailed level instance features to the RoI head, RoI aligns pooler resolution is increased from 14x14 to 35x35, meaning that a larger resolution of 35x35 input to the final classification and regression head represents every shortlisted proposal. With this change, the final classification and regression layers get region embedding with more detailed representations.

E. Modifications in Region Proposal Network (RPN) Architecture

Experiments are also conducted to validate whether extending the default RPN layers helps improve the performance. Specifically, ablation studies are carried out by introducing different architectural blocks in RPN before the 1x1 convolutional layers for predicting the objectness and localization scores. A simple bottleneck block with 2 convolutional layers is used, which reduces the total channels by a factor of 2 and then increases the channels by 2. Using squeeze and excitation block [29], we append the squeeze and excitation branch on top of the residual connection. The squeeze and excitation branch is shown in Fig. 6.

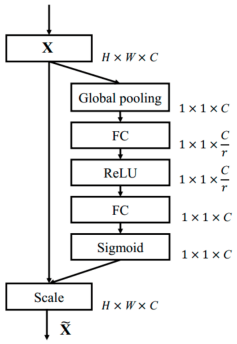

Fig. 6: Squeeze and excitation block used in RPN

F. Pre-training techniques

We explore different pre-training techniques when transferring a model for the iSAID object detection task. The baseline model uses an imageNet-1k pre-trained ResNet backbone. We study if a pre-trained model already tailored for an object detection task is more suitable for the downstream detection task. To validate this, we study the effect of using a Faster-RCNN model, which is firstly trained on the COCO detection dataset. Additionally, we explore how a model trained on a segmentation task performs on iSAID dataset. As segmentation is a more dense prediction task, its trained model will more aggressively attend to even small highlights in the images. To check the performance of the pre-trained segmentation model, we use the Mask-RCNN model trained on the COCO instance segmentation dataset. When transferring the model for the detection task, the segmentation mask head is removed, and the rest of the model is used as a detector.

G. Data Augmentations

We use Data Augmentation at different scales to address the problem of high inter-class and intra-class variations. The baseline considers a single scale of 800 (shorter side). Instead, we use scale augmentations at six different scales (1200,1000,800,600,400). Using scaling at such extreme scales accounts for category instances of varying scales.

H. Pre & Post Processing techniques

Anchor Box Sizes: The region proposal network uses anchor box techniques to find possible class-agnostic object proposals. The default baseline configuration for size anchor boxes for each feature map is tailored for object detection for natural images. However, as shown in Fig. 7, the iSAID dataset contains high size variations with big and small objects. Therefore, for a similar proportion of objects in anchor boxes, two new anchor box size configurations are added on top of the default one. Additionally, 16x16 and 384x384 resolution anchor boxes are provided for all feature maps at three different aspect ratios (0.5, 1.0, 1.5).

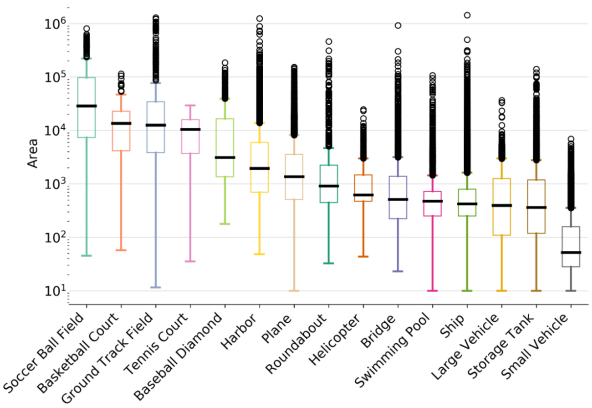

Fig. 7: Box plot showing the high size variations of different objects present in the iSAID dataset. Plot adapted from [4]

Non-maximum suppression thresholds at training and testing: Object instance count per image for the iSAID dataset is very high i-e up to 200 annotations can be present in a single image. This number is very high compared to the instance count per image in natural images such as COCO. Thus, we propose increasing RPN's top proposals during training and testing time. Specifically, we increase the training and testing time RPN proposals from 2000 and 1000 to 3000, which helps to increase the overall recall rate of the detector.

IV. Experiments and Results

Evaluation details: For our experiments, we train all models on the iSAID official training set and report results on the official iSAID validation set. In addition, for every evaluation, we report the standard MS-COCO evaluation metrics to compare the performance of models. Specifically, we are interested in the AP (averaged across different IoU thresholds) and AP_{50}.

Training details: For all experiments, we use the detectron2 framework [30]. For backbone experiments, we use learning rates of 0.0025 and 0.00025. We observe that the ConvNext and SWIN backbone are very sensitive to learning rate, and thus to stabilize training, we use a lower learning rate of 0.0025. For the rest of the experiments, we fix the learning rate to 0.0025. All models are trained for 100k iterations with a batch size of 2. Experiments with a Deformable-ResNet backbone take roughly 8 hours on a single Quadro RTX 6000 24GB GPU. For the standard ResNet backbone, a single training time is approximately 7 hours.

Choosing the best backbone: For the initial set of experiments, we use a learning rate of 0.00025. The quantitative results are shown in Table I. The SWIN-Base and ConvNext-Base models provide degraded performance compared to the baseline ResNet-101 model. SWIN backbone provides 29.11% AP and 53.11% AP50, while ConvNext provides 28.35% AP and 51.45% AP50. Baseline ResNet-101 backbone provides 32.47% and 55.47% AP and AP50, respectively. The Deformable-ResNet backbone provides the highest AP and AP50 of 32.60% and 56.35%, respectively. Further, the baseline and Deformable-ResNet models are trained with a scaled learning rate of 0.0025 in Table I. Scaled learning rate results for SWIN-Base and ConvNext-Backbone are unavailable because of unstable training in this setting. The Deformable ResNet backbone exceeds the performance of the baseline model by 1.36% AP and 1.65% AP50. For the experiments in the subsequent sections, we fix the backbone with the ResNet Deformable backbone.

Choosing better transfer-learning technique: With the chosen backbone as Deformable-ResNet-101, we ablate on using different pre-trained models for iSAID object detection tasks. Table II shows the performance results of different pre-trained models. All experiments use the Deformable-ResNet backbone. It can be observed from the results that using completely pre-trained models perform better than using the model with only the backbone pre-trained. Specifically, Faster-RCNN with COCO-detection pre-trained model provides 41.86% AP

Backbone	LR	AP	AP50	AP75	APs	APm	APl
ResNet 101(baseline)	2.5e-4	32.47	55.47	33.98	19.93	39.8	40.13
SWIN Base	2.5e-4	29.11	53.11	28.44	19.43	36.32	33.63
ConvNext Base	2.5e-4	28.35	51.45	28.23	17.57	35.73	30.87
Deformable Resnet101	2.5e-4	32.6	56.35	33.19	20.24	39.53	38.10
Using higher learning rate							
ResNet 101(baseline)	2.5e-3	38.68	61.38	41.73	24.54	46.21	51.14
Deformable Resnet 101	2.5e-3	**40.04**	**63.02**	**43.02**	**25.87**	**47.36**	**51.60**

TABLE I: Effect using different backbones with Faster-RCNN detector. Deformable ResNet101 backbone provides improvement over the baseline model as compared to others. Using higher learning rate for SWIN and ConvNext results in unstable training, thus they are not included.

and 63.28% AP50. Interestingly, a Mask-RCNN pre-trained model on the COCO segmentation model provides further improved performance of 42.10% AP and 63.34% AP50. Therefore, we use the COCO-Segmentation pre-trained model and the Deformable-ResNet backbone for our next series of experiments.

Model	Pre-training	AP	AP50	AP75	APs	APm	APl
Deformable R-101	ImageNet 1k	40.04	63.02	43.02	25.88	47.37	51.60
Deformable R-101	COCO Det	41.86	63.29	**46.09**	26.11	48.29	57.76
Deformable R-101	COCO Seg	**42.11**	**63.35**	45.98	**26.36**	**49.46**	**57.81**

TABLE II: Effect of using different transfer-learning techniques. Generally, pre-training provides better results—pre-training using the COCO segmentation dataset results in better performance than the detection pre-training.

Performance evaluation of different loss functions: After selecting the suitable pre-trained model and backbone, we study the use of different loss functions on top of the previous best model. The results are shown in Table III. When the GIoU loss is used, it performs better than the baseline model by providing 42.70% AP and 63.64%. Using federated loss does not help much in achieving gain on top of the baseline instead reduces the accuracies slightly. Specifically, it provides 42.03% AP and 63.64% AP50. The use of focal loss also shows degraded results providing 17.62% AP and 25.50% AP50, respectively. The combination of losses also resulted in less performance than the baseline results. The use of focal and federated loss has been studied for datasets with a considerable number of classes, which is likely to be the reason for not showing improved results for the iSAID dataset, where there are only 15 categories. We add the GIoU loss in our best model configurations from these experiments and report subsequent experiments on the best model configuration achieved so far.

Results of architectural changes in RPN: Table IV shows the effect of using different blocks instead of the default 3x3 single convolution layer in RPN. From the results, both variants having squeeze and excitation block, as well as the residual bottleneck block, show less performance than the baseline. The Squeeze and Excitation block provides 41.91% AP and 63.23% AP50, while the residual block provides

Loss	AP	AP50	AP75	APs	APm	APl
Baseline*	42.11	63.35	45.98	**26.36**	49.46	57.81
GIOU	**42.70**	**63.95**	**47.08**	26.28	**50.60**	**58.53**
Only Federated	42.03	63.65	46.18	26.12	49.38	56.54
Federated + GIOU	41.86	63.29	46.09	26.11	48.29	57.76
Focal Loss	17.62	25.50	19.56	11.20	20.65	18.55
Focal + GIOU	13.78	19.41	15.16	8.53	16.53	15.79

TABLE III: Shows the effect using different losses. GIOU loss achieves higher accuracy over the baseline in each metric except APs. * Here, the baseline is ResNet101 deformable with MS COCO segmentation pre-training.

37.101% AP and 54.93% AP50 accuracies show that extending the RPN network by stacking new blocks and layers generally does not help. Based on these experiments, we do not include these modifications as they do not provide clear improvements over the current best model.

Method	AP	AP50	AP75	APs	APm	APl
Baseline*	42.70	63.95	47.08	26.28	50.60	58.53
Baseline + Residual	37.10	53.93	41.55	20.12	46.30	56.97
Baseline + SQ Net	41.91	63.23	45.79	25.65	50.26	57.74

TABLE IV: Ablations on using residual and squeeze and excitation blocks in RPN. No clear results were observed by extending the RPN network. * Baseline here is ReseNet101 deformable backbone with MS COCO segmentation pre-training and GIOU loss.

Pre & post processing and RoI pooling: The results using different pre-post processing techniques and increasing RoI pooling are summarized in Table V. Adding new anchor box sizes decreases 0.45% AP, and it increases AP50 by 0.42% AP. Increasing pooler resolution only helps in increasing the APs. By combining all proposed techniques, the model provides 42.735 AP and 65.16% AP50 indicating improvements over its default version. We have also conducted detailed ablation experiments on the effect of using different pooler resolutions and anchor box sizes. These results are shown in Table VIII and IX for pooler resolution and anchor box sizes, respectively. These experiments use the original default baseline (vanilla Faster-RCNN with FPN (R-101) and imageNet-1k pre-training) with no other changes. We have observed that combining the best modifications sometimes does not provide better performance. Additionally, increasing the NMS from its default values of 2000 to 3000 resulted in higher performance. Therefore, we use this improved model with all the modifications mentioned above for the subsequent experiments.

Adding data-augmentation: As our final modification, resizing training images with extreme scales, including 1200 and 400 scales, helped improve the model. On top of our modernized model (configuration in the last section), the results after adding the scale augmentation are shown in Table VI. Specifically, the model achieves 43.12% AP and 65.84% AP50. It also improves the other metrics' performance except AP small (APs).

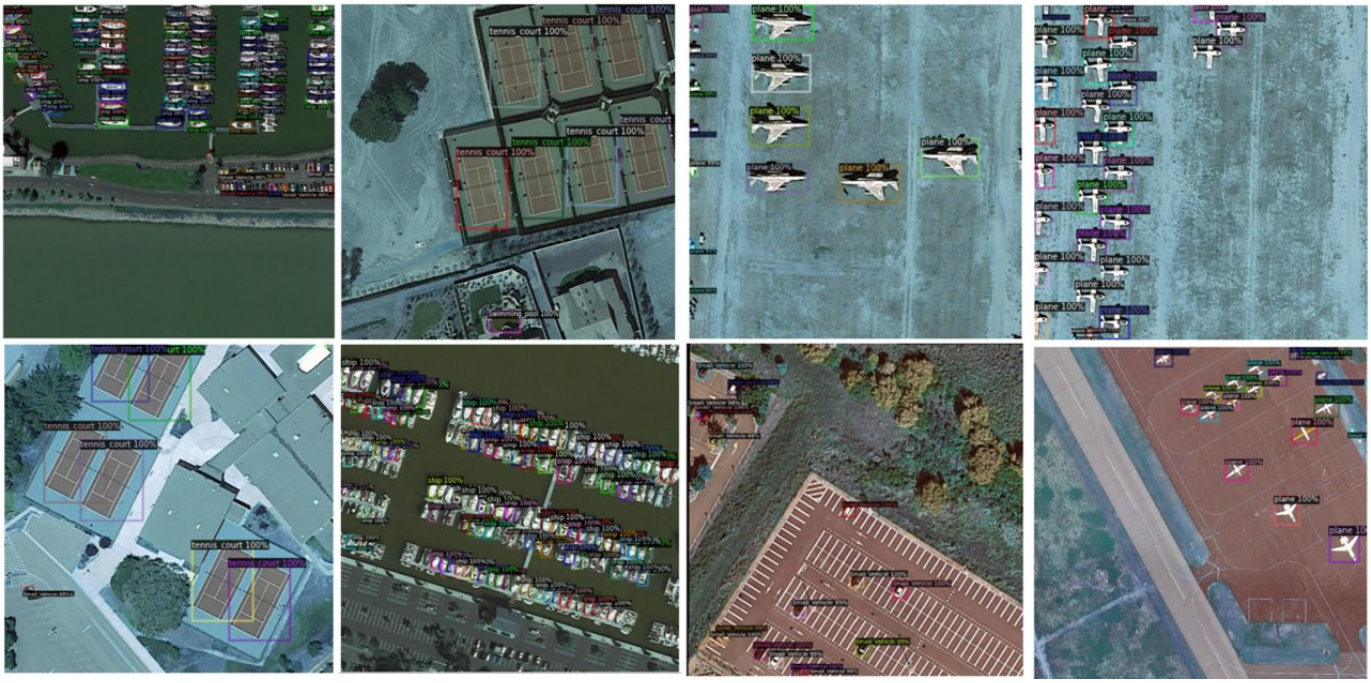

Fig. 8: shows qualitative results of detection on the iSAID validation set. The model is able to accurately detect objects with varying scales.

Method	AP	AP50	AP75	APs	APm	APl
Baseline*	42.70	63.95	**47.08**	26.28	**50.60**	**58.53**
+ anchor-box sizes	42.24	64.37	45.27	26.98	48.78	57.38
+ pool-resolution (35x35)	41.99	63.87	45.82	**26.77**	48.21	57.13
+ NMS techniques	**42.73**	**65.161**	46.146	27.69	49.644	57.05

TABLE V: Effect of introducing additional anchor boxes sizes, increased pooling resolution and NMS. * Baseline here is ReseNet-101 deformable backbone with MS COCO Segmentation pre-training and GIOU loss.

Method	AP	AP50	AP75	APs	APm	APl
Baseline*	42.73	65.16	46.14	**27.69**	49.64	57.05
+ Resizing at different scales	**43.12**	**65.84**	**46.98**	27.20	**49.74**	**58.15**

TABLE VI: Effect of using image augmentation at different scales. * Baseline here is ReseNet101 deformable backbone with MS COCO segmentation pre-training, GIOU loss and pre-post processing techniques.

Modernized Faster-RCNN for Aerial OD: Here we summarize the final proposed model with all the positive modifications that helped improve the model's performance. Compared to the default vanilla Faster-RCNN with R-101 FPN backbone and imageNet-1k pre-training, our model uses the Deformable-R-101 backbone with the COCO-Instance segmentation pre-training. The training objective uses GIoU loss. In the RPN, we increase the anchor box sizes proportion, and increased pooler resolution (35x35) is used in the RoI Pooler block. In addition, the NMS topk proposals at training and test time are increased to 3000, and data augmentations with resizing at different scales are also incorporated. Results for these modifications on top of the default baseline are summarised in Table VII. The qualitative results for the final model are shown in Fig. 8.

Method	AP	AP50	AP75	APs	APm	APl
Basline	38.68	61.38	41.73	24.54	46.21	51.14
+ Deform-R-101 backbone	40.05	63.02	43.02	25.88	47.37	51.60
+segmentation pre-train	42.11	63.35	45.98	26.36	49.46	57.80
+GIOU Loss	42.70	63.95	**47.08**	26.28	**50.60**	**58.53**
+ anchor + pooler + NMS	42.73	65.16	46.15	27.69	49.64	57.05
+ Aug (resizing)	**43.12**	**65.84**	46.98	**27.20**	49.74	58.15

TABLE VII: Summary of the effect of each positive proposed modification. Results of proposed model on iSAID test set are provided in the last row.

Unuseful Modifications: Given the considerable improvement of the final proposed model over the baseline, several modifications were expected to yield better performance but provided sub-optimal results. Using SWIN and ConvNext backbone could only push the performance a little. The full potential of SWIN and ConvNext backbone can be utilized using scaled hyperparameters and multi-GPU training, such as the batch size of 16 and high learning rate. However, it could not be covered here due to the limited resources. Additionally, for the losses, the focal and federated loss provided poor results. For the focal loss, it is better at handling class imbalance datasets and requires more classes compared to only 15 classes present in the iSAID dataset. For the federated loss, it requires optimal weight assignment. Experiments of extending RPN layers did not help and resulted in redundant model

parameters; this shows that stacking new blocks and layers to increase the model complexity only sometimes helps. Overall, studying each change's effect was useful when modernizing a vanilla baseline architecture.

V. Conclusion

In this work, we studied using a two-stage object detector, Faster-RCNN, for object detection on the iSAID dataset. Notably, we observed that the vanilla Faster-RCNN is sub-optimal when directly detecting objects in satellite images. We found that one of the primary reasons for that is the domain shift of the dataset. Vanilla Faster-RCNN is designed for natural images, which are very different from satellite images. To tailor the detector model to perform better on satellite images, we proposed several modifications over various blocks of the model and conducted a detailed set of experiments. Specifically, performance gains over the baseline model were achieved by proposing changes in the backbone, pre-training, loss function, RoI Pooler, and different pre & post-processing techniques. Additionally, we observed that extending the model layers in a detector, such as in RPN, only sometimes helps to increase the performance. Overall, the proposed model provides significant improvements over the vanilla baseline model and verifies the importance of each proposed component introduced.

References

[1] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digital Signal Processing*, p. 103514, 2022. 1

[2] J. Han, J. Ding, J. Li, and G.-S. Xia, "Align deep features for oriented object detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2021. 1

[3] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "Dota: A large-scale dataset for object detection in aerial images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3974–3983, 2018. 1, 2

[4] S. Waqas Zamir, A. Arora, A. Gupta, S. Khan, G. Sun, F. Shahbaz Khan, F. Zhu, L. Shao, G.-S. Xia, and X. Bai, "isaid: A large-scale dataset for instance segmentation in aerial images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 28–37, 2019. 1, 2, 5

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, pp. 886–893, Ieee, 2005. 1

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015. 1, 2

[7] S. M. Azimi, E. Vig, R. Bahmanyar, M. Körner, and P. Reinartz, "Towards multi-class object detection in unconstrained remote sensing imagery," in *Asian Conference on Computer Vision*, pp. 150–165, Springer, 2018. 1

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014. 1

[9] 2015. 1

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016. 1

[11] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017. 1

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018. 1

[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020. 1

[14] I. Ševo and A. Avramović, "Convolutional neural network based automatic object detection on aerial images," *IEEE geoscience and remote sensing letters*, vol. 13, no. 5, pp. 740–744, 2016. 2

[15] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, "D2det: Towards high quality object detection and instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11485–11494, 2020. 2

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 2

[17] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian, "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 370–386, 2018. 2

[18] L. W. Sommer, T. Schuchert, and J. Beyerer, "Fast deep vehicle detection in aerial images," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 311–319, IEEE, 2017. 2

[19] F. Yang, H. Fan, P. Chu, E. Blasch, and H. Ling, "Clustered object detection in aerial images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8311–8320, 2019. 2

[20] K. Li, G. Wan, G. Cheng, L. Meng, and J. Han, "Object detection in optical remote sensing images: A survey and a new benchmark," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 159, pp. 296–307, 2020. 2

[21] D. Lam, R. Kuzma, K. McGee, S. Dooley, M. Laielli, M. Klaric, Y. Bulatov, and B. McCord, "xview: Objects in context in overhead imagery," *arXiv preprint arXiv:1802.07856*, 2018. 2

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. 3

[23] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 764–773, 2017. 3

[24] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021. 3

[25] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," *arXiv preprint arXiv:2201.03545*, 2022. 3

[26] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017. 4

[27] X. Zhou, V. Koltun, and P. Krähenbühl, "Probabilistic two-stage detection," *arXiv preprint arXiv:2103.07461*, 2021. 4

[28] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 658–666, 2019. 4

[29] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018. 4

[30] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." https://github.com/facebookresearch/detectron2, 2019. 5

Appendix

Here we provide additional ablation experiments with Faster-RCNN with ResNet101-deformable backbone. Table VIII shows the effect of introducing additional anchor boxes sizes and Table IX shows the effect on performance by increasing the pooler resolution.

Method	AP	AP50	AP75	APs	APm	APl
Basline (Deform-R-101)	40.05	63.02	43.02	**25.88**	47.37	51.60
Pooler Resolution 14	39.92	**63.18**	42.72	24.97	47.57	51.86
Pooler Resolution 24	**40.16**	62.86	**43.21**	24.75	**47.62**	52.37
Pooler Resolution 34	39.47	62.46	42.09	24.07	47.18	**52.63**

TABLE VIII: Effect on performance by increasing the pooler resolution. The results indicate that pooler resolution provides better performance solely in some of metrics but overall provides sub-optimal results.

Method	AP	AP50	AP75	APs	APm	APl
Basline (Deform-R-101)	40.05	63.02	43.02	**25.88**	47.37	51.60
Add 16x16 anchor to P2	40.42	63.56	**43.86**	25.42	47.94	52.71
Add 384x84 Anchor to P5	40.12	63.04	43.07	24.56	47.77	52.37
Add 16 and 384 to P2 and P5	**40.51**	**64.00**	43.48	25.75	47.65	**52.88**

TABLE IX: Effect of introducing 16, 384 anchor box resolutions to P2 and P5 respectively.