IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 25, 2022, accepted May 5, 2022, date of publication May 10, 2022, date of current version May 17, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3174087

Improving Small and Cluttered Object Detection by Incorporating Instance Level Denoising Into Single-Shot Alignment Network for Remote Sensing Imagery

YASSIN ZAKARIA[1], SAHAR A. MOKHTAR[1], HODA BARAKA[2], AND MAYADA HADHOUD[2]
[1]Electronics Research Institute, Giza 11843, Egypt
[2]Department of Computer Engineering, Faculty of Engineering, Cairo University, Giza 12613, Egypt

Corresponding author: Yassin Zakaria (yassin.zakaria@eri.sci.eg)

ABSTRACT Object detection on satellite and aerial images has gained the attraction of the scientific community of computer vision due to its immense value, high difficulty, and the development of large-scale datasets enough to train deep learning models. The progress is tremendous with the increase in the precision of the fast single-stage object detection models which used to sacrifice precision for speed. Aerial and satellite images are of large sizes which makes slow models infeasible for production. Single-shot Alignment Network (S2A-Net) is a fast and competitive single-stage model in terms of precision. However, there is a potential to increase its precision in detecting small and cluttered objects in a complex background. In this paper, a new hybrid approach by incorporating Instance Level Denoising (ILD) module from Small, Cluttered, and Rotated Object Detector ++ (SCRDet++) into S2A-Net is proposed. The model was trained and tested on Dota V1.0. The proposed model achieves a higher mean average precision (mAP) than S2A-Net to be 79.73% and when it was trained using the Kullback–Leibler divergence as a regression loss function, the proposed model can reach as high as 80.39% mAP.

INDEX TERMS Convolutional neural network (CNN), deep learning, object detection, remote sensing.

I. INTRODUCTION

Modern aerial and satellite imaging provides an abundance of images that present a great opportunity for many applications such as urban planning, geographic information systems, and intelligent transportation systems. Object detection on aerial and satellite images can be used in those applications by extracting crucial information. For example, capturing urban hotspots, which is done using GPS data [1], can use information extracted by object detection. Object detection on aerial and satellite images is a challenging task because those images are characterized by low resolution, large size, and bird-eye view. With the recent breakthroughs in object detection using deep learning, it became a logical step to apply deep learning to remote sensing imagery. However, it is not straightforward to adapt object detection models, designed for natural images, to remote sensing images.

The associate editor coordinating the review of this manuscript and approving it for publication was Bilal Alatas.

In computer vision, object detection extracts a set of either axis-aligned or oriented bounding boxes that locate objects of interest in an image. Oriented bounding boxes (OBB) are often the target for detection in satellite and aerial images because of the huge intersection over union (IoU) between axis-aligned boxes due to object rotation. This makes assigning boxes to objects and post-processing harder for axis-aligned bounding boxes.

This paper is focused on deep learning approaches in an attempt to utilize the capability of deep learning to generalize on many different object detection situations compared to handcrafted features and template matching methods. For a more comprehensive analysis and a discussion on other approaches, this paper [2] presents an extensive survey on classical methods.

Due to the characteristics of remote sensing images, the objects are often small, cluttered, and in arbitrary orientations. Basic CNN with standard convolutional layers aggregates the features using filters on regular grids to form deep features which causes inter-class feature coupling and

blurred noisy features. For a fast single-stage object detector to be accurate, a feature alignment module and a feature decoupling module for denoising are needed. Single-shot Alignment Network (S2A-Net) [3] is a fast and accurate single-stage object detector that uses an Alignment Convolution Layer (AlignConv) for feature alignment. Small, Cluttered, and Rotated Object Detector ++ (SCRDet++) [4] uses Instance Level Denoising (ILD) for feature decoupling and denoising.

This paper's contributions are as follows:

- Proposing a hybrid model for object detection by incorporating Instance Level Denoising into Single-shot Alignment Network to improve the detection of small and cluttered objects. This model will be referred to as S2A-Net-ILD.
- Improving the training of the hybrid model by using the Kullback–Leibler divergence (KLD) [5] regression loss function to achieve a more competitive result of 80.39% mAP. This model will be referred to as S2A-Net-ILD-KLD.

The rest of the paper is organized as follows: Section 2 reviews the related works in a brief manner and the models used to create the proposed hybrid model in more detail. Section 3 details the proposed model and the reasons for its inception alongside the implementation details. Section 4 describes the used dataset, details the experimental setup, and presents the results including a comparison with recent works and a discussion of the results. Finally, conclusions and future work are given in Section 5.

II. RELATED WORK

Object detection in deep learning is often a supervised machine learning task where the model is trained on images with bounding boxes' labels. In machine learning techniques, a two-stage object detection model consists of (1) region proposal and feature extraction; (2) object classification, localization, and refinement for each region. The most influential two-stage deep learning object detection model is Faster R-CNN [6]. Deep learning facilitated the development of single-stage methods but they might sacrifice precision for speed. YOLO [7], SSD [8], and Retinanet [9] are examples of single-stage models.

Object detection usually exhibits performance degradation when faced with object occlusion, varying illumination, small objects, object closeness, object rotation, and varying objects' aspect ratio. These issues are significant in aerial and satellite images where most objects are extremely small, rotated, close to each other, and placed within a complex background. In addition, aerial and satellite images are large so slow models are not a viable option for many applications. A basic deep convolutional neural network will not generalize well on this task because it is only translation equivariant.

The aforementioned challenges have instigated the development of several techniques. To address the small object issue, Feature Pyramid Network (FPN) [10] has been used extensively in the field with [3]–[5], [11]–[28] using it. FPN integrates features from multiple layers including deep and shallow features to create rich hierarchical features for object detection. Thus, the models have sufficient features to train for small object detection. Li *et al.* [12] created an enhanced FPN (eFPN) which incorporates a sub-inception block while integrating between different levels of features. Small, cluttered, and rotated objects detector (SCRDet) [29] created its own feature fusion network called sampling and feature fusion network (SF-Net) which incorporates its own inception block for fusing features from multiple layers. AProNet model enhances the multi-scale features from FPN by fusing it with the output from AProNet's feature enhancement module. AProNet's feature enhancement module is based on Atrous Spatial Pyramid Pooling (ASPP) [30] to extract geometric features from the input image using Atrous/Dilated separable convolutions. Multi-scale data augmentation in training and testing is also used alongside FPN to facilitate generalization. Another approach was to employ contextual features such as holistic image features [15], [31], features pooled from a zoomed-out region [14], [20] or features pooled by neighborhood aggregation [32]. Those approaches try to enhance the features of the object by incorporating contextual features.

To address the high variance in object rotation, which causes multiple issues in prediction formulation, both single-stage and two-stage models handle it differently. In two-stage models, Rotational Region Convolutional Neural Network (R2CNN) [32] used axis-aligned anchor boxes. The features are then pooled from the enclosing axis-aligned box. The generated region of interest (RoI) features are forwarded to the second stage in which the oriented bounding boxes are predicted. The prediction strategy of R2CNN was used in [4], [12], [27], [29]. In contrast, the Rotational Region Proposal Network (RRPN) [33] uses more anchors with different angles and the features are pooled from the oriented box. Rotation RoI pooling layer (RRoI) [33] and rotated RoI Align [34] are examples of pooling layers to extract features from OBB proposals. Some recent models [11], [13], [16], [35] use rotated anchor boxes. RoI (Region of Interest) transformer model [14] created two new layers: Rotational RoI learner and Rotated Position Sensitive RoI Align (RPS-RoI-Align). Rotational RoI learner uses axis-aligned anchors to produce horizontal proposals which are refined into OBB proposals, then RPS-RoI-Align pools the features from the OBB produced by the RoI learner. Thus, the RoI transformer reduced the required number of anchor boxes by refining the horizontal bounding boxes proposal into oriented ones. ReDet [19] extended the work of the RoI transformer. ReDet uses rotation-equivariant ResNet (ReResNet) with rotation-equivariant feature pyramid network (ReFPN) as a backbone based on e2cnn model [36] to generate rotation-equivariant features. Thus, Rotated RoI (RRoI) pooling methods alone is not sufficient, as orientation alignment is needed for rotation-equivariant features to transform into rotation-invariant features for the

final prediction phase. ReDet introduced a Rotation-invariant RoI Align layer to pool from rotation-equivariant features to generate rotation-invariant features for the prediction. While RoI-transformer-based models decrease the potential of overfitting by decreasing the number of needed anchors, they are slow and increase the number of parameters in the model due to the need to predict horizontal proposals and then transform them into oriented ones. Both Oriented R-CNN [26] and DODet [28] directly predict oriented proposals using horizontal anchor boxes. Thus, they alleviate the downsides of the RoI transformer. Region proposal network in Oriented R-CNN uses midpoint offset representation [26] to generate its OBB proposals. DODet opted to predict the aspect ratio and area of the OBB instead of the width and the height directly. The DODet model uses the OBB prediction from the second stage for RoI pooling for the classification. While this potentially slows down the model, the features are aligned better for the classifier.

For single-stage models, features are not pooled out of the feature map but the feature refinement stage has been applied in [3], [17] to align features and reconstruct them for the final prediction stage using the initial prediction. This enables the single-stage models to refine features before the final prediction without sacrificing speed by pooling features out for a second stage like in the two-stage methods. Refined single-stage rotation detector (R3Det) [17] uses the initial OBB predictions to determine the corners and the center of an object in the feature map. Then it concatenates those points with the initial feature map to generate enhanced features. Single-shot Alignment Network (S2A-Net) [3] uses Alignment Convolution Layer to align features using a deformable convolution operation rather than standard 2D convolution.

Other works explored changing the loss function to incorporate rotation in the prediction. The default regression loss function from object detection is smooth L1-loss which suffers from boundary discontinuity and a square-like problem [18] when rotation regression is introduced. The common objective metric used to evaluate object detectors is mean average precision (mAP). Thus, the goal of the model is to have high recall and precision. The OBB output from the model should have the correct class label and have a high intersection over union (IoU) with the ground truth box. Thus, IoU-smooth L1-loss was first used by the Small, Cluttered, and Rotated Object Detector (SCRDet) model [29]. IoU-smooth L1-loss weights the smooth L1-loss by the skew IoU between the predicted output and the ground truth box. Therefore, the loss function and objective function are more aligned. The AProNet model predicts the angle by regressing two values $(apro_y, apro_x)$ such that the angle equals $atan(apro_y/apro_x)$ and $(apro_y, apro_x) = (sin(\theta)\times w, cos(\theta)\times w))$. This representation is free of the angular periodicity issue. AProNet uses a weighted smooth L1-loss between the predicted angle and the ground truth where the weight is a function of the angular difference. Xue Yang *et al.* [18] proposed the Gaussian Wasserstein Distance regression loss function (GWD). The GWD loss formulates the oriented bounding boxes as Gaussian distributions. Thus, the Gaussian Wasserstein Distance between the two oriented bounding boxes can be measured. Xue Yang *et al.* [5] proposed using Kullback–Leibler divergence (KLD) instead of GWD as a loss function due to being scale-invariant for objects of different sizes. Yang Xue and Yan Junchi [23] proposed using Circular Smooth Labels (CSL) to transform the rotation prediction task from regression to classification to prevent the discontinuous boundaries problem from emerging. Then Yang Xue *et al.* [24] proposed Densely Coded Labels (DCL) to shorten the code length to lighten the weight of the angular rotation prediction layer while using Angle Distance and Aspect Ratio Sensitive Weighting (ADARSW) to improve the performance of DCL-based models by increasing their sensitivity to the angular distance and the object's aspect ratio.

To address the issue of small objects in a complex background, some approaches seek to differentiate between objects and backgrounds on a pixel level. The problem can be transformed into instance segmentation as in [25], [37]. A post-processing step is then applied to the generated instance segmentation to produce OBB. Another approach is to learn an attention module to score feature importance either in a completely unsupervised manner [22], [28], [31] or guided by a supervised semantic segmentation task [4], [12], [25], [29], [37]. Finally, semantic segmentation can be learned as a complementary task to enhance the performance and its features can be used as inputs for the other object detection modules as in [12], [25], [37]. Those approaches are not mutually exclusive.

Attention modules can be categorized into spatial, channel-wise, and self-attention. Spatial attention reweights features based on their spatial location. Channel-wise attention reweights channels' importance based on the input features. Self-attention is an attention mechanism that enables long-range dependencies between features. CAD-Net [31] and Li *et al.* [12] use spatial attention. CenterMap [25] and SCRDet [29] use both spatial and channel-wise attention. CG-Net [22] uses multi-head self-attention where each feature vector is a whole channel slice to decrease the computational overhead of multi-head self-attention. DODet [28] uses rotated discriminative pooling (RDP) which is inspired by discriminative RoI pooling [38] to extract RoI for classification. RDP uses adaptive weighted pooling [38] to weight different features within the extracted RoI in an unsupervised manner.

In this paper, a hybrid model is proposed that modifies S2A-Net [3] by adding Instance Level Denoising (ILD) [4] to overcome the problem of detecting small and cluttered objects within a noisy background. Therefore, the next 2 subsections will give more details on SCRDet++ and S2A-Net. Both models start with a backbone model with FPN for feature extraction.

A. SMALL, CLUTTERED, AND ROTATED OBJECT DETECTOR ++ (SCRDet++)

SCRDet++ [4] is a two-stage object detector. The model architecture extracts the features from FPN, then an Instance Level Denoising (ILD) module is applied on each feature level to enhance them before the detection phases. ILD is an attention module to score feature importance which is trained in a supervised manner by learning semantic segmentation on top of the attention feature map. The goal of ILD is to decouple the features of the objects of different categories into their respective channels and suppress noisy features. The rest of the model is similar to R2CNN [32] which uses horizontal anchor boxes but with RoI warping layer [39] and the new IoU-smooth L1 loss function. The whole model can be seen in Fig.1.

ILD module takes the input feature map and applies a sequence of dilated convolutions [40] to increase the respective field for the coarse semantic segmentation task. Those dilated convolutions are considered feature enrichment convolutional layers. The gradient from learning the coarse semantic segmentation task guides the attention module. This happens because the attention map is generated from the same input features used for the coarse semantic segmentation task. ILD is shown in Fig.2. ILD applies an element-wise product between the attention map and the original coupled feature map. The equation for ILD operation in inference is shown in equation (1) such that $\mathbf{X}$ is the input feature map. CNN_{ILD} is the ILD's convolutional layers. $\odot$ is the Hadamard product.

$$ILD(\mathbf{X}) = CNN_{ILD}(\mathbf{X}) \odot \mathbf{X} \tag{1}$$

B. SINGLE-SHOT ALIGNMENT NETWORK (S2A-Net)

S2A-Net [3] is a fast single-stage model with a refinement stage using feature alignment while also being computationally efficient. Single-shot in S2A-Net does not refer to one-shot learning. S2A-Net introduced two modules for refinement called Feature Alignment Module (FAM) and Oriented Detection Module (ODM).

Feature Alignment Module (FAM) is used to refine the input features. FAM does the initial oriented bounding boxes (OBB) prediction which is used to refine the input features. Horizontal anchor boxes are used for the initial OBB prediction. A single horizontal anchor design is used for computational efficiency. Alongside regression predictions, FAM generates a probability score for an object of a certain class being contained within the anchor box. The regression output is used to calculate the offset field for the new convolutional layer called Alignment Convolution Layer. This can be seen in Fig.3. Alignment Convolution Layer (AlignConv) is a new type of deformable convolutional layer [41] that samples features for the convolutional operation using the offset field rather than using the standard grid-based sliding window on the feature map as shown in Fig.4. The classification output is discarded during inference but it exists in training to guide the training process. FAM refines features to be an input for the ODM.

The following equations represent the operations of AlignConv on the 2-D plane. Equation (2) shows an example where *REG* is the standard regular grid for 3×3 ($k = 3$) standard convolution with dilation rate 1. Equation (3) is the standard convolution operation, while equation (4) is the AlignConv operation. $\mathbf{W}$, $\mathbf{P}$, and $\mathbf{X}$ are the convolution filter weights, the center location of the convolution operation, and the input feature map respectively. O is the offset field which is calculated in equation (5). An anchored bounding box is represented by (x, y, w, h, θ). $R(\theta)$ is the rotation matrix. S is the stride between adjacent locations on the feature map $\mathbf{X}$ when mapped back to the input image.

$$REG = \{(-1,-1),(-1,0),\ldots,(0,1),(1,1)\} \tag{2}$$

$$Y(\mathbf{P}) = \sum_{\mathbf{reg}\in REG} \mathbf{W}(\mathbf{reg}) \cdot \mathbf{X}(\mathbf{P}+\mathbf{reg}) \tag{3}$$

$$Y(\mathbf{P}) = \sum_{\mathbf{reg}\in REG;\mathbf{o}\in O} \mathbf{W}(\mathbf{reg}) \cdot \mathbf{X}(\mathbf{P}+\mathbf{reg}+\mathbf{o}) \tag{4}$$

$$\mathbf{L}_{\mathbf{P}}^{\mathbf{reg}} = \frac{1}{S}((x,y) + \frac{1}{k}(w,h) \cdot \mathbf{reg} \cdot R(\theta)^{\mathsf{T}})$$

$$O = \{\mathbf{L}_{\mathbf{P}}^{\mathbf{reg}} - \mathbf{P} - \mathbf{reg}\}_{\mathbf{reg}\in REG} \tag{5}$$

Oriented Detection Module (ODM) applies active rotating filters (ARF) [42] on the feature map produced by FAM. ARF is a convolutional filter which encodes orientation information in its output by rotating its filter N times to produce N+1 features. Those features encode N+1 different orientations to help the bounding box regression task as both the features and the task are orientation-sensitive. The equation for ARF operation for i-th orientation output is shown in equation (6). $\mathbf{Y}$ is the output of ARF in which its channels consist of $\mathbf{Y}^i$ for all values of i. $\mathbf{W}_{\theta_i}^n$ is the n-th orientation channel for the rotated filter at θ_i while $\mathbf{X}^n$ is the input feature map at orientation channel n.

$$\mathbf{Y}^i = \sum_{n=0}^{N-1} \mathbf{W}_{\theta_k}^n \cdot \mathbf{X}^n, \quad \theta_i = i\frac{2\pi}{N}, \quad i = 0,\ldots,N-1 \tag{6}$$

For the classification task, a rotation-invariant feature is preferred. Thus, the feature map, which is produced from ARF, is forwarded to an oriented response pooling layer. This layer extracts the features corresponding to the orientation with the highest response value as shown in equation (7) where $\hat{\mathbf{X}}$ is the output after the pooling operation. The output feature map from the oriented response pooling layer is forwarded to the classification branch.

$$\hat{\mathbf{X}} = \max\ \mathbf{X}^n, \quad 0 < n < N-1. \tag{7}$$

The rest of the model is the normal OBB prediction using two branches one for classification and the other for regression. The notable difference in training is the new addition to the loss function for training FAM as a refinement module for a single-stage detector. The whole network is shown in Fig.5.

III. PROPOSED MODEL

Both SCRDet++ and S2A-Net follow different design decisions to solve object detection on aerial and satellite images.

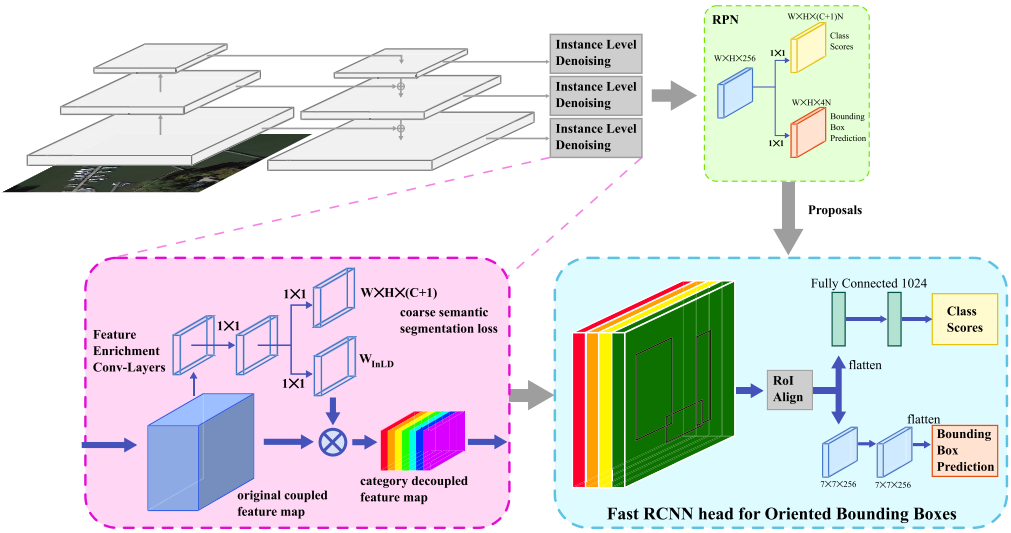


FIGURE 1. SCRDet++ model.

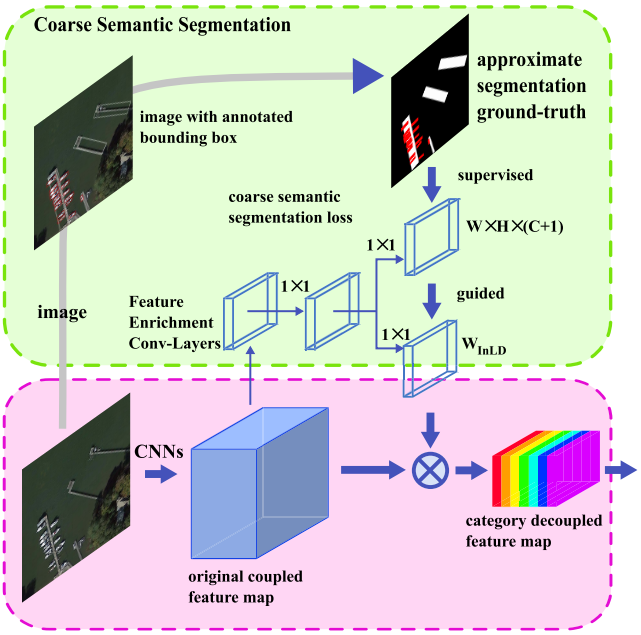


FIGURE 2. Instance level denoising module from SCRDet++ [4].

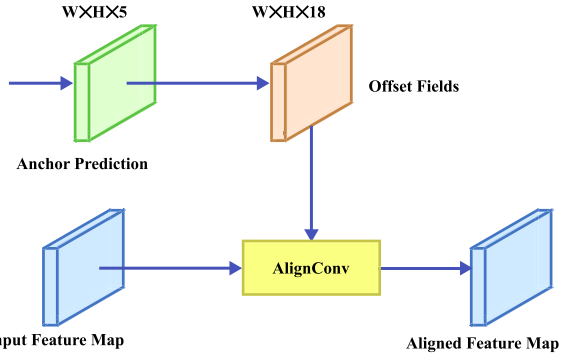
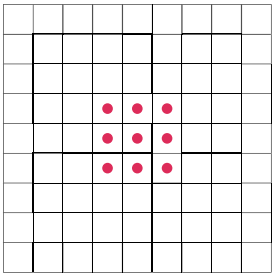


FIGURE 3. S2A-Net feature alignment module in inference.

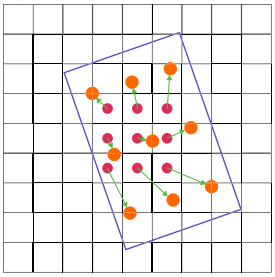
Those design decisions help the model tackle different difficulties faced in applying deep learning to object detection on aerial and satellite images. The goal of the proposed method is to create a hybrid model which combines the strength of the two models without compromising model efficiency. SCRDet++ deals with background noise and object cluttering with its attention module called Instance Level Denoising (ILD). SCRDet++ augments the object detection task by learning an approximate semantic segmentation task on top of the task. S2A-Net handles feature misalignment due to rotation by FAM module for single-stage detectors and improves prediction on rotated objects by ODM module. Both models handle different yet major problems with object detection on aerial and satellite images. The proposed method creates a hybrid approach between the two models by incorporating an ILD module into S2A-Net before the FAM module to enhance features by feature denoising. Thus, the new model can handle the clutter of objects and suppress noisy features from the background before feature alignment. ILD is applied before FAM to enhance the initial regression prediction and prevent cascading of noisy background features to later modules. Our model is shown in Fig.6. Table.1 shows the theoretical justification for incorporating ILD into S2A-Net. ILD has been selected as the attention module over the self-attention used in CG-Net [22], as long-range dependencies are not desirable for detecting small and cluttered objects. This can be shown in the low result of 73.23% AP for small vehicles for CG-Net.

A. IMPLEMENTATION DETAILS

Implementation details will be divided into 4 subsections. The first subsection gives details about the neural network structure module-wise. The second subsection formulates how the prediction is done. The third subsection specifies



(a) Standard 2D convolution.



(b) AlignConv on OBB.

FIGURE 4. Convolution operations.

TABLE 1. Theoretical advantages of the proposed model.

Method	Model type	Handles noise feature	Handles Object rotation
SCRDet++	Two-stage	**Yes**	No
S2A-Net	Single-stage	No	**Yes**
Our method	Single-stage	**Yes**	**Yes**

the loss function used. The last subsection describes the post-processing procedure.

1) MODULES

The proposed neural network consists of a backbone model for feature extracting, Instance Level Denoising (ILD), Feature Alignment Module (FAM), and Oriented Detection Module (ODM).

a: BACKBONE MODEL FOR FEATURE EXTRACTING

We use Retinanet [9] architecture with pre-trained ResNet-50 [43] as the backbone for feature extraction. Retinanet uses FPN to produce deep hierarchical features suitable for object detection on satellite and aerial images. FPN produces 5 levels of hierarchical features. Those features are then used in the prediction modules of FAM and ODM after getting operated on by the ILD module. Backbone model ResNet-50 is a middle ground deep learning model which can be fine-tuned on a consumer-grade machine while not compromising much on the model performance. ResNet models use skip connections and batch normalization [44] which helps in training very deep neural networks. In our work, we freeze the weights of the first two blocks to prevent the training in editing those weights, as early convolutional layers of Resnet produce useful low-level features. All batch normalization layers in the pre-trained backbone are frozen as well.

b: INSTANCE LEVEL DENOISING (ILD)

After extracting features from the feature pyramid network, each feature map is input to ILD to enhance it by attention. ILD operation starts with applying feature enrichment convolutional layers to the input feature map. The feature enrichment convolutional layers consist of a sequence of dilated convolutions: four 3×3 dilated convolutions with a dilation rate of 2 then followed by one 3×3 dilated convolutions with a dilation rate of 4. Those dilated convolutions increase the respective field on the feature. After the dilated convolutions, a 1×1 convolution is applied to combine channel features. Thus, the features are enhanced for the segmentation task. The ReLU [45] activation function is used. The enhanced features are input to two branches. The first branch is to predict the segmentation mask by a 1×1 convolution followed by softmax and the second branch is to produce an attention map by a 1×1 convolution. For the second branch, no activate function is used. Sigmoid can be used but it hinders the gradient flow in the training process. The attention map produced is also of the size of the input feature and the attention process is an element-wise multiplication between the two feature maps. This module is based on the official GitHub repository implementation for SCRDet++ (FPN-based) [46] retrieved in December 2020.

c: FEATURE ALIGNMENT MODULE (FAM)

FAM takes the output of ILD to refine those features by an Alignment Convolution Layer which takes dense regression boxes prediction to calculate offset fields. Then the convolution operation uses the offset fields rather than the standard grid. Regression box predictions are created by applying a sequence of two 3×3 convolution layers followed by a one 1×1 convolution layer on the input of FAM. Alongside the regression box prediction, there is a classification branch with a similar architecture to regression box prediction to compute the class probability of the boxes. The ReLU activation function is used except at the regression and classification output of the FAM.

d: ORIENTED DETECTION MODULE (ODM)

The feature map generated from the FAM is the input to the Oriented Detection Module (ODM).An Active Rotating Filter layer (ARF) is applied to the input feature with the configuration of 1 orientation, 8 different rotations, and a 3×3 kernel. The 8 rotations cover angles from 0 to 360 degrees. The feature map produced from ARF is forwarded to the regression branch in the prediction phase. The output of ARF is forwarded to an oriented response pooling layer, then it is forwarded to the classification branch in the prediction phase. The architecture of the prediction phase is similar to the one used in the FAM but its output is the final prediction of the model. The ReLU activation function is used except at the outputs.

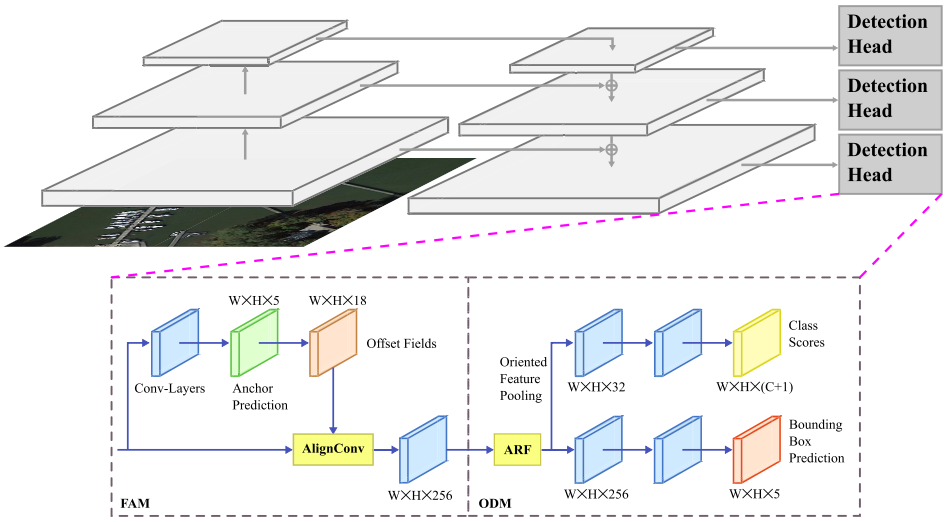


FIGURE 5. S2A-Net model.

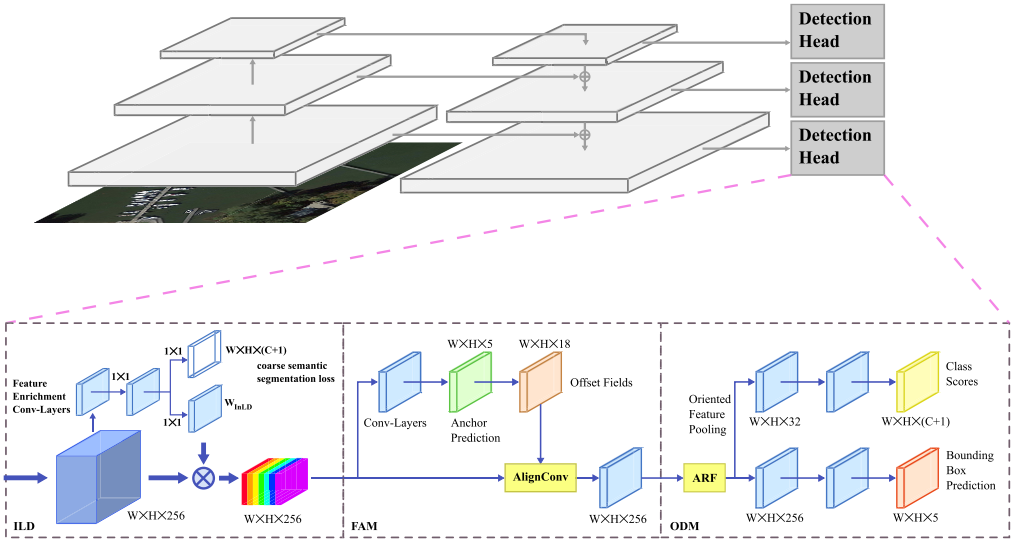


FIGURE 6. Our hybrid model S2A-Net with instance level denoising.

2) PREDICTIONS

a: ANCHOR PREDICTION

The regression output formulation can be expressed in equation (8). The regression output of FAM predicts the difference (Δ) between the ground truth box and the anchor box. The ground truth box is represented by the box center $\mathbf{CP}_g = (x_g, y_g)$, the width w_g, the height h_g, and the angle θ_g, where the anchor box is denoted by its center point $\mathbf{CP}_a = (x_a, y_a)$, width w_a, height h_a, and angle θ_a. As all anchor boxes are horizontal then $\theta_a = 0$ and the rotation matrix $R(\theta_a)$ is the identity matrix. K is an integer.

$$\begin{aligned} \Delta\mathbf{CP}_g &= (\mathbf{CP}_g - \mathbf{CP}_a)R(\theta_a)(1/w_a, 1/h_a) \\ (\Delta w_g, \Delta h_g) &= (\ln(w_g/w_a), \ln(h_g/h_a)) \\ \Delta\theta_g &= \frac{1}{\pi}(\theta_g - \theta_a + K\pi) \end{aligned} \tag{8}$$

For the ODM regression output, it is similar to FAM but instead of using the default square anchor boxes as a reference, it uses the decoded output from FAM.

Along with each regression output from the neural network, there is a classification output which classifies the type of object including the background to represent non-objects.

b: ANCHOR MATCHING CRITERIA

Skew IoU is used to assign ground truth boxes with an anchor box. If the IoU between the object and the anchor box is greater than the foreground threshold then the ground truth box is assigned to the anchor as long as it has the maximum IoU value than the rest of the ground truth boxes. Thus, the anchor box is given a positive label. The anchor box can also be assigned to a negative label if the IoU between any ground

truth box with it is below the background threshold. If the anchor box is not assigned with either a negative or a positive label, it is ignored in the training process. Last but not least, there is a corner case of assigning a ground truth box to an anchor with maximum IoU value between them when ground truth boxes could not be assigned to any anchor box. This is needed due to the huge diversity in aspect ratios of certain categories to realistically design anchor boxes that cover them (e.g. Bridges).

3) LOSS FUNCTIONS

The ILD module generates coarse semantic segmentation which needs to be trained in a supervised manner. Thus, the segmentation output loss (mask loss) function is the average of the pixel-wise softmax cross-entropy loss function as in equation (9). L_{ILD} in equation (10) is the pixel-wise softmax cross-entropy where c is the number of classes including the background. λ_{ILD} is the weight of the segmentation loss function. h and w are the height and width of the segmentation mask. $\mathbf{O}_{ij}^{m}$ is the output from the ILD and $\mathbf{O}_{ij}^{g}$ is the ground truth.

$$mask\ Loss = \frac{\lambda_{ILD}}{(w \times h)}(\sum_{i}^{h}\sum_{j}^{w} L_{ILD}(\mathbf{O}_{ij}^{m}, \mathbf{O}_{ij}^{g})) \quad (9)$$

$$L_{ILD}(\mathbf{O}^{m}, \mathbf{O}^{g}) = -\sum_{c}(\mathbf{O}_{c}^{g} log(\mathbf{O}_{c}^{m})) \quad (10)$$

For the ODM and the FAM regression loss, the smooth-L1 loss is used and for the ODM and the FAM classification losses, the focal loss is used. The whole loss function can be seen in equation (11) where equation (12) is for smooth-L1 and equation (13) is for focal loss. λ_{FAM} and λ_{ODM} are the weights of the regression and classification loss functions for FAM and ODM respectively. N_{PF} and N_{PO} are the numbers of positive samples in the FAM and ODM respectively. L_c is the focal loss with α and γ as weighting hyperparameters, which control the importance of misclassified examples in the overall loss. L_r is the smooth-L1 loss. P_i^* is a flag that is equal to 1 if the sample is positive and 0 otherwise. This ensures that the regression is only trained when there is an object within the anchor. $\mathbf{C}_i^F$ and $\mathbf{C}_i^O$ are the classification output from FAM and ODM. C_i^g is the classification ground-truth label. $\mathbf{r}_i^F$ and $\mathbf{r}_i^O$ are the regression output from FAM and ODM. r_i^g is the regression ground truth. C and r are general terms for classification and regression outputs in this context.

$$\begin{aligned} Loss = {} & \frac{\lambda_{ILD}}{(w \times h)}(\sum_{i}^{h}\sum_{j}^{w} L_{ILD}(\mathbf{O}_{ij}^{m}, \mathbf{O}_{ij}^{g})) \\ & + \frac{\lambda_{FAM}}{N_{PF}}(\sum_{j} L_c(\mathbf{C}_i^F, \mathbf{C}_i^g) + \sum_{j} P_i^* L_r(\mathbf{r}_i^F, \mathbf{r}_i^g)) \\ & + \frac{\lambda_{ODM}}{N_{PO}}(\sum_{j} L_c(\mathbf{C}_i^O, \mathbf{C}_i^g) + \sum_{j} P_i^* L_r(\mathbf{r}_i^O, \mathbf{r}_i^g)) \end{aligned} \quad (11)$$

$$L_r(\mathbf{r}, \mathbf{r}^g) = \sum_{i} Smooth_{L1}(\mathbf{r}_i^g - \mathbf{r}_i)$$

$$Smooth_{L1}(r) = \begin{cases} |r| & \text{if } |r| > 1 \\ 0.5(r)^2 & \text{otherwise} \end{cases} \quad (12)$$

$$L_c(\mathbf{C}, \mathbf{C}^g) = \sum_{c} -\alpha(1 - P(\mathbf{C}_c, \mathbf{C}_c^g))^{\gamma} \ln(P(\mathbf{C}_c, \mathbf{C}_c^g))$$

$$P(C, C^g) = \begin{cases} C & \text{if } C^g = 1 \\ 1 - C & \text{otherwise} \end{cases} \quad (13)$$

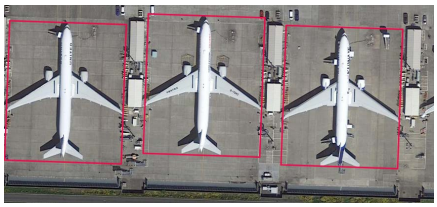
Kullback–Leibler divergence (KLD) loss function [5] can be used as a regression loss instead of the smooth-L1 loss. According to [5], the KLD loss function has some desired properties over the smooth-L1 loss such as the KLD loss is more correlated to the IoU between the predicted OBB and the ground truth OBB than the smooth-L1 loss. For objects with high aspect ratios, a small angle regression error in object detection translates into a large localization error. The KLD loss function weights those errors higher than the smooth-L1 loss. Also, the KLD loss does not suffer from the square-like problem, unlike the smooth-L1 loss where the smooth-L1 loss will be significantly higher for correctly localized square-like objects due to the periodicity of the angle. The KLD loss is scale-invariant with the OBB size, so the gradients from the loss function for errors in localizing small objects are not dominated by the gradients for errors in larger objects. This enables the KLD loss to train models for high-precision OBB detection.

The KLD loss considers both the ground truth boxes and the predicted boxes as Gaussian distributions. Therefore, the KLD distance can be calculated between the predicted OBB and the ground truth OBB. The formulation of an OBB as Gaussian distribution is shown in equation (14) and Fig.7 where the distribution $\mathcal{N}(\mu, \Sigma)$ has the mean μ and the covariance Σ. Mean μ is the box's center point CP. $R(\theta)$ is the rotation matrix.

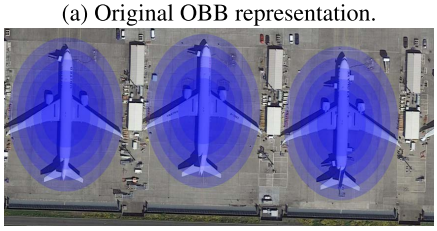
$$\begin{aligned} \mu &= \mathbf{C}\mathbf{P}^{\intercal} = (x, y)^{\intercal} \\ \Lambda &= \begin{pmatrix} w/2 & 0 \\ 0 & h/2 \end{pmatrix} \\ \Sigma^{1/2} &= R(\theta)\Lambda R(\theta)^{\intercal} \end{aligned} \quad (14)$$

While the KLD loss function operates on the Gaussian distribution representation of an OBB, the regression outputs of the CNN are still encoded as the difference between the ground truth OBBs and the anchor boxes. This encoding helps in training neural networks as it is translation invariant and normalizes the expected output of the CNN around a mean of zero. The regression outputs of the CNN are transformed first into Gaussian distribution representation before applying the KLD loss function. The KLD distance is shown in equation (15). Tr is the trace function.

$$\begin{aligned} D_{KLD}(\mathcal{N}_p||\mathcal{N}_g) = {} & \frac{1}{2}((\mu_p - \mu_g)^{\intercal}\Sigma_g^{-1}(\mu_p - \mu_g) \\ & + Tr(\Sigma_g^{-1}\Sigma_p) + \ln\frac{|\Sigma_g|}{|\Sigma_p|}) - 1 \end{aligned} \quad (15)$$



(a) Original OBB representation.



(b) 2-D Gaussian distribution representation.

FIGURE 7. Oriented bounding boxes represented as 2-D Gaussian distributions for the KLD loss function.

The KLD loss function is shown in equation (16). If the KLD distance $D_{KLD}(\mathcal{N}_p||\mathcal{N}_g)$ is equal to zero then the loss function will be equal to zero as well, while if the KLD distance is large, the loss function will be equal to 1. The ln function is used on the KLD distance to smooth out the loss function.

$$L_{KLD}(\mathcal{N}_p, \mathcal{N}_g) = 1 - \frac{1}{1 + \ln(1 + D_{KLD}(\mathcal{N}_p||\mathcal{N}_g))} \tag{16}$$

4) POST-PROCESSING

For post-processing, the prediction output of the model is forwarded to non-maximum suppression (NMS) to filter duplicate boxes. The non-maximum suppression threshold is set to be 0.1 IoU between boxes for filtering and the maximum number of output boxes is 2000.

IV. EXPERIMENTAL RESULTS

This section has 3 subsections. The first subsection describes Dota V1.0 [47] dataset used to evaluate the proposed model. The second subsection details the experimental setup including the used machine and software, the dataset preparation and the data augmentation, and the hyperparameters. The third part is for the results. This includes an ablation study comparing the baseline model with the proposed model, followed by a comparison between the proposed model with a selection of state-of-the-art methods on the Dota V1.0 dataset.

A. DATASET

There are different datasets for the task of object detection in remote sensing images. Different benchmarks have been used in different researches to evaluate different models. In this paper, Dota V1.0 [47] will be used due to the following factors:

- It is a public dataset of aerial images from different elevations and different resolutions.
- The dataset contains 2,806 images. The images in the dataset are divided into 1/2 for training, 1/6 for validation, and 1/3 for testing.
- It has 188,282 annotated objects which can be enough to train a deep learning model.
- It has a public leaderboard.
- It has oriented bounding box annotations for the objects.
- It has images from different resolutions ranging from 800×800 to 4000×4000.
- It covers 15 different categories which are Plane (PL), Baseball diamond (BD), Bridge (BR), Ground track field (GTF), Small vehicle (SV), Large vehicle (LV), Ship (SH), Tennis court (TC), Basketball court (BC), Storage tank (ST), Soccer-ball field (SBF), Roundabout (RA), Harbor (HA), Swimming pool (SP), and Helicopter (HC).

While there are more recent versions for the Dota dataset, Dota V1.0 has been used the most for evaluation even by the most recent works in the literature. Evaluating existing models, using the most recent version of the Dota dataset, is left for future work.

The evaluation metric used is mean average precision (mAP) and average precision (AP) for each category.

B. EXPERIMENTAL SETUP

1) THE USED MACHINE AND SOFTWARE

All models are trained and tested on a machine equipped with a 24 GB VRAM NVIDIA GeForce RTX 3090 GPU and with Ubuntu 20.04 as the operating system. The deep learning framework used to code the models is Pytorch [48] with MMDetection [49].

2) DATASET PREPARATION AND DATA AUGMENTATION

The dataset is prepared in two different ways depending on the experiment type: a setup for the ablation study for the hyperparameters and a setup for the comparison with the State-of-the-Art.

During the ablation study for the hyperparameters: the model is trained on the training dataset only and the validation dataset is used to test the model. Single-scale images are used during training and testing. The images are then cropped by a sliding window of 1024×1024 with a stride of 824. The images with sizes less than 1024×1024 are zero-padded. This setup is used to decrease the required training time to search for the hyperparameters.

During comparison with the state-of-the-art: the model is trained on both the training and the validation dataset. The test dataset is used for the test. Multi-scale images are used in training and testing the model where the images in the dataset are resized by 0.5, 1, and 1.5 times the original scale while maintaining the aspect ratio. The images are then cropped by a sliding window of 1024×1024 with a stride of 512. The images with sizes less than 1024×1024 are zero-padded.

During training for both setups, we augment the dataset by flipping and rotating the images where the images are

TABLE 2. Results on the DOTA V1.0 test dataset task 1 (on oriented bounding boxes) and comparison between baseline S2A-Net with our model. S2A-Net-ILD-KLD is our model trained using the KLD loss function.

Methods	PL	BD	BR	GTF	SV	LV	SH	TC	BC	ST	SBF	RA	HA	SP	HC	mAP	FPS
S2A-Net Baseline	88.89	83.60	57.74	**81.95**	79.94	83.19	89.11	90.78	84.87	87.81	70.30	68.25	78.30	77.01	69.58	79.42	**16.0**
S2A-Net-ILD (Ours)	**89.23**	82.88	55.33	80.30	80.76	83.51	**89.30**	90.69	**87.34**	87.72	**72.92**	67.38	78.45	78.95	71.27	79.73	12.5
S2A-Net-ILD-KLD (Ours)	88.25	**85.12**	**58.24**	80.21	**80.82**	**83.80**	89.06	**90.78**	85.86	**88.15**	69.39	**69.70**	**78.85**	**80.00**	**77.69**	**80.39**	12.5

randomly flipped horizontally and vertically and randomly rotated by one of the following angles: 30, 60, 90, 120, and 150 degrees.

3) HYPERPARAMETERS

The following are the optimizer hyperparameters and the training scheme:

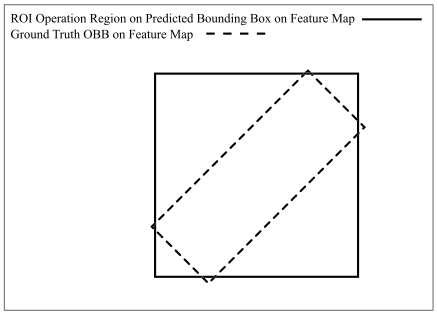
- Stochastic gradient descent (SGD) with momentum optimizer is used with an initial learning rate of 0.01 with momentum 0.9 and weight decay 0.0001.
- Linear learning rate warm-up is used for the first 500 iterations.
- Gradient clipping is used.
- Batch size of 8 is used.
- Model is trained for 12 epochs.

The selected optimizer hyperparameters are adopted from the S2A-Net paper. The model suffers from overfitting when trained for more epochs. This might be a result of using a high initial learning rate of 0.01 with a batch size of just 8. Increasing both the batch size and the number of epochs might improve the model training. Also, decreasing the learning rate while increasing the number of epochs can improve the model training. Thus, exploring those possibilities is left for future work. We initially tested ADAM [50] and RMSProp [51]. The exploding gradient problem emerged while using any of them unless a very low learning rate is used. Potential approaches to prevent exploding gradients are to set up different update rates for early layers which involves more trial and error or to apply gradient clipping while tuning the weight decay hyperparameter to prevent steep updates to the model weights. Therefore, it is left for future work. A learning rate warm-up with gradient clipping was only needed for SGD with momentum to prevent exploding gradients from emerging.

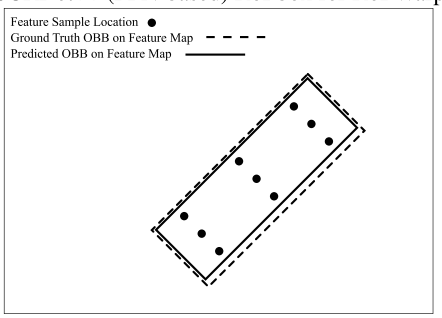
The following are the loss function hyperparameters used to train the model:

- The weight of the classification and the regression losses for ODM and FAM are 1.0 while the weight of the segmentation loss is 0.1.
- $\alpha = 0.25$ and $\gamma = 2.0$ for the focal losses.

The model uses a single squared anchor per location for each feature level with a scale of 4 times the stride size of the level. The stride sizes are 8, 16, 32, 64, and 128. Thus, the scales of the anchors are 32, 64, 128, 256, and 512, respectively. Therefore, higher levels of the FPN are used to detect larger objects. The anchor matching IoU thresholds are 0.5 or higher for positive matching and 0.4 or less for negative matching.



(a) SCRDet++ (FPN-based) RoI box for RoI Warping



(b) Feature Alignment Module (FAM) Operation

FIGURE 8. RoI warping in SCRDet++ based on [46] and alignment convolution operation from FAM in S2A-Net. If both regression box predictions are correct, RoI warping can propagate more noisy features around the object of interest for the final prediction phase.

TABLE 3. Results on the DOTA v1.0 validation dataset including usage of different segmentation loss function weights (λ_{ILD}) for our model S2A-Net with ILD. The models are trained on the training dataset only.

Model	λ_{ILD}	λ_{FAM} , λ_{ODM}	mAP
S2A-Net	0	1 , 1	71.86
S2A-Net-ILD	0	1 , 1	69.64
S2A-Net-ILD	0.1	1 , 1	**72.48**
S2A-Net-ILD	0.2	1 , 1	71.96
S2A-Net-ILD	0.5	1 , 1	70.797
S2A-Net-ILD	1	1 , 1	68.66

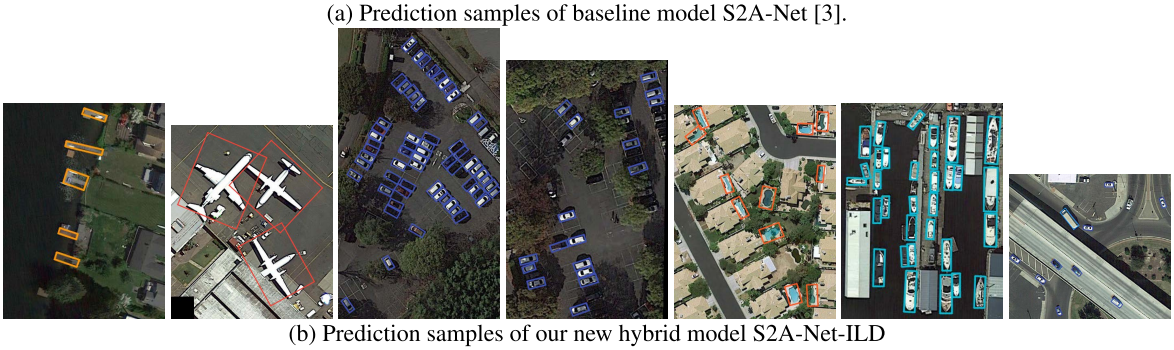
C. RESULTS

1) ABLATION STUDY

The ablation study consists of 3 parts:

1) Part 1 is a comparison between the proposed model (S2A-Net-ILD) against the baseline S2A-Net while

(a) Prediction samples of baseline model S2A-Net [3].



(b) Prediction samples of our new hybrid model S2A-Net-ILD

FIGURE 9. Result samples on some of the improved categories from DOTA V1.0. The prediction samples of baseline model S2A-Net is generated by using official implementation of S2A-Net [52].

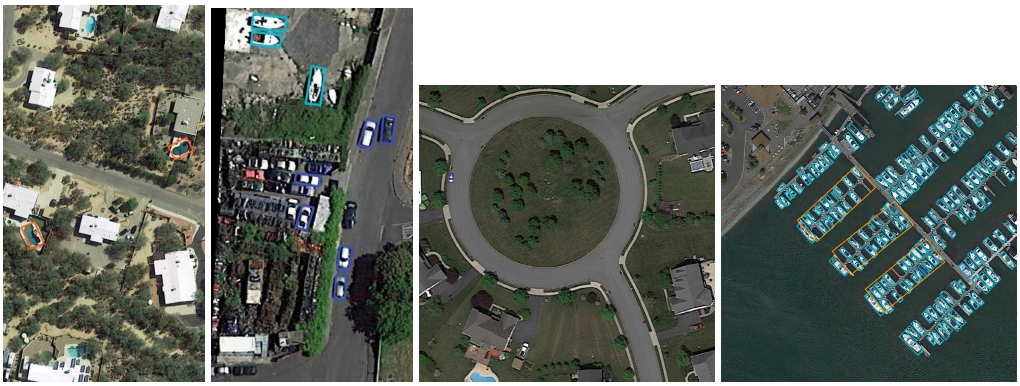
TABLE 4. Results on the DOTA V1.0 test dataset task 1 (on oriented bounding boxes). The models are trained using data augmentation and multi-scale training. The R3Det-GWD results are reported without the ensemble method for a fair comparison. * means best results overall. ' means best results for a single-stage model.

Methods	Backbone	PL	BD	BR	GTF	SV	LV	SH	TC	BC	ST	SBF	RA	HA	SP	HC	mAP
Two-stage Models																	
RoI-Transformer [14]	R-101-FPN	88.64	78.52	43.44	75.92	68.81	73.68	83.59	90.74	77.27	81.46	58.39	53.54	62.83	58.93	47.67	69.56
CAD-NET [31]	R-101-FPN	87.8	82.4	49.4	73.5	71.1	63.5	76.7	90.9	79.2	73.3	48.4	60.9	62.0	67.0	62.2	69.9
SCRDet [29]	R-101-FPN	90.18	81.88	55.30	73.29	72.09	77.65	78.06	90.91	82.44	86.39	64.53	63.45	75.77	78.21	60.11	75.35
CenterMap [25]	R-101-FPN	89.83	84.41	54.60	70.25	77.66	78.32	87.19	90.66	84.89	85.27	56.46	69.23	74.13	71.56	66.06	76.03
CSL [23]	R-152-FPN	90.25	**85.53***	54.64	75.31	70.44	73.51	77.62	90.84	86.15	86.69	69.60	68.04	73.83	71.10	68.93	76.17
SCRDet++ [4]	R-101-FPN	90.05	84.39	55.44	73.99	77.54	71.11	86.05	90.67	87.32	87.08	69.62	68.90	73.74	71.29	65.08	76.81
Li et al. [12]	R-101-FPN	**90.41***	85.21	55.00	78.27	76.19	72.19	82.14	90.70	87.22	86.87	66.62	68.43	75.43	72.70	57.99	76.36
CG-Net [22]	R-101-FPN	88.75	85.18	57.41	71.88	73.23	82.68	88.14	**90.90***	86.00	85.37	62.99	66.74	77.98	79.90	71.27	77.89
AProNet [27]	R-101-FPN	88.77	84.95	55.27	78.40	76.65	78.54	88.45	90.83	86.56	87.01	65.62	70.29	75.43	78.17	67.28	78.16
Faster R-CNN OBB RoI-Transformer [24]	R-50-FPN	87.89	85.01	57.83	78.55	75.22	84.37	88.04	90.88	87.28	85.79	71.04	69.67	79.00	**83.27***	73.43	79.82
ReDet [19]	ReR50-ReFPN	88.81	82.48	60.83	80.82	78.34	**86.06***	88.31	90.87	**88.77***	87.03	68.65	66.90	79.26	79.71	74.67	80.10
DODet [28]	R-50-FPN	89.96	85.52	58.01	81.22	78.71	85.46	88.59	90.89	87.12	87.80	70.50	**71.54***	82.06	77.43	74.47	80.62
Oriented R-CNN [26]	R-50-FPN	89.84	85.43	**61.09***	79.82	79.71	85.35	88.82	90.88	86.68	87.73	72.21	70.80	**82.42***	78.18	74.11	**80.87***
Single-Stage Models																	
R3Det [17]	R-152-FPN	89.80	83.77	48.11	66.77	78.76	83.27	87.84	90.82	85.38	85.51	65.67	62.68	67.53	78.56	72.62	76.47
R3Det-DCL [24]	R-152-FPN	89.26	83.60	53.54	72.76	79.04	82.56	87.31	90.67	86.59	86.98	67.49	66.88	73.29	70.56	69.99	77.37
R3Det-GWD [18]	R-50-FPN	88.43	84.33	56.91	**82.19***	76.69	83.23	86.78	88.90	83.93	85.73	72. 07	65.67	76.76	78.37	65.31	78.35
	R-152-FPN	89.28	83.70	**59.26'**	79.85	76.42	83.87	86.53	89.06	85.53	86.50	**73.04***	67.56	76.92	77.09	71.58	79.08
R3Det-KLD [5]	R-50-FPN	89.90	84.91	59.21	78.74	78.82	83.95	87.41	89.89	86.63	86.69	70.47	70.87	76.96	79.40	78.62	80.17
	R-152-FPN	**89.92'**	**85.13'**	59.19	81.33	78.82	**84.38'**	87.50	89.80	87.33	87.00	72.57	71.35	77.12	79.34	**78.68***	**80.63'**
S2A-Net [3]	R-101-FPN	89.28	84.11	56.95	79.21	80.18	82.93	89.21	**90.86'**	84.66	87.61	71. 66	68.23	78.58	78.20	65.55	79.15
	R-50-FPN	88.89	83.60	57.74	81.95	79.94	83.19	89.11	90.78	84.87	87.81	70.30	68.25	78.30	77.01	69.58	79.42
S2A-Net-ILD (Ours)	R-50-FPN	89.23	82.88	55.33	80.30	80.76	83.51	**89.30***	90.69	**87.34'**	87.72	72.92	67.38	78.45	78.95	71.27	79.73
S2A-Net-ILD-KLD (Ours)	R-50-FPN	88.25	85.12	58.24	80.21	**80.82***	83.80	89.06	90.78	85.86	**88.15***	69.39	**69.70'**	**78.85'**	**80.00'**	77.69	80.39

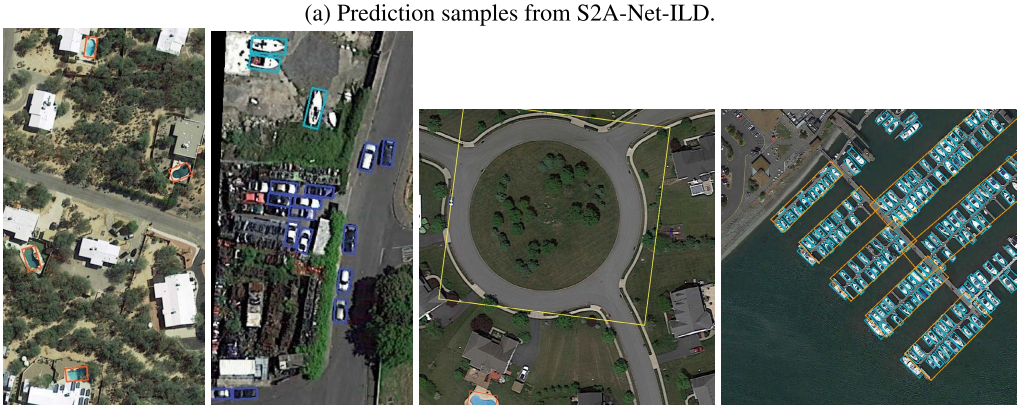
using the same regression loss function (smooth L1-loss).

2) Part 2 is an ablation study on the selection of the segmentation loss (λ_{ILD}) weight.
3) Part 3 is a comparison between the proposed model trained using the KLD loss function (S2A-Net-ILD-KLD) against those trained using the smooth L1-loss (S2A-Net-ILD)

Part 1: The addition of ILD only to S2A-Net improved the mAP by 0.31% as shown in Table.2. The new model (S2A-Net-ILD) offers advantages and disadvantages over the baseline. The advantages include: a general trend in improving object detection for vehicles like planes, small vehicles, large vehicles, and helicopters by 0.34%, 0.82%, 0.32%, 0.19% and 1.69% AP as those objects are often in clusters or within a complex background and shading. The model also improves the detection of small objects/areas surrounded by a complex background like swimming pools by 1.94%. The highest detection improvements are for both Soccer-ball fields and Basketball courts by 2.62% and 2.47% AP. We don't have a theoretical hypothesis for the increase of AP for Soccer-ball fields and Basketball courts. Detection of harbors has slightly improved by 0.15% AP, as some harbors are small objects and often accompanied by docking ships.



(a) Prediction samples from S2A-Net-ILD.



(b) Prediction samples from S2A-Net-ILD-KLD

FIGURE 10. **Result samples on some of the improved categories from DOTA V1.0 while using the KLD as a regression loss function.**

Fig.9 shows the improvement in the results of our method against the baseline. The new model suffers from degradation in the detection of larger objects that are more dominant and not within clusters of other objects like bridges, roundabouts, baseball diamonds, and ground track fields by -0.78%, -2.41%, -1.65%, and -0.87% mAP. The results suggest that ILD is unnecessary and hinders S2A-Net to perform on larger objects especially when there are few observations to train the model on. For example, the instance counts of ground track fields and roundabouts in the dataset are 678 and 871 while there are 48,891 observations of small vehicles. Instance level denoising (ILD) also adds computation cost on top of S2A-Net and slows it down from 16 FPS to 12.5 FPS on RTX 3090. S2A-Net using a backbone of ResNet 101 with FPN runs at 12.7 FPS under the same conditions, so our model is comparable in speed with it but with better overall mAP as shown in Table.4.

The performance gain from ILD is not as significant in the new model as in adding it to the baseline of SCRDet++ (FPN-based). This can be explained by the feature sampling and alignment methods in both models. SCRDet++ uses RoI warping on the RoI generated by the region proposal network (RPN) while S2A-Net uses the Alignment Convolution Layer on the predictions generated from FAM. The RoI generated from RPN in SCRDet++ are horizontal bounding boxes around the object projected on the feature map, unlike the Alignment Convolution Layer which applies deformable convolution by sampling features from the oriented bounding boxes. Thus, noisy features will cascade to the final prediction phase even with correct RPN predictions in SCRDet++ while not as significantly in S2A-Net. This is shown in Fig.8. Thus, the effect of using ILD to denoise and decouple features in SCRDet++ is more significant than S2A-Net. Applying ILD on S2A-Net still improves the prediction of small and cluttered objects because:

- Applying ILD before the FAM can improve the initial regression prediction of the bounding boxes for small and cluttered objects which is used for feature alignment.
- FAM applies Alignment Convolution Layer operation on feature areas where there is not an object as the classification branch is not used during testing to suppress the results. Thus, applying ILD to decrease background noise can decrease false positives.

Part 2: Table.3 shows the effect of changing the loss function weights. When the weight of the segmentation loss (λ_{ILD}) is set to 0, the ILD module is trained alongside the rest of the network without further supervision. When λ_{ILD} has a non-zero value, gradients from coarse semantic segmentation affect and guide the training of the ILD module.

The results suggest that coarse semantic segmentation is useful for training the ILD module. Without it, the model performance degrades. The model performs worse with high λ_{ILD} values such as 0.5 and 1. The goal of the model is object detection rather than coarse semantic segmentation, so high λ_{ILD} values cause the gradients from coarse semantic segmentation to be dominant.

Part 3: As shown in Table. 2, using the KLD loss significantly improves the performance of the proposed model by 0.66% mAP to reach 80.39% mAP. Using the KLD loss improves the detection of many object categories with large aspect ratios such as bridges, and harbors by 2.91%, and 0.40% AP. It also improves the detection of small objects and areas such as small vehicles, and swimming pools by 0.06%, and 1.05% AP. Using the KLD loss improves detection for square-like objects such as baseball diamonds, storage tanks, and roundabouts by 2.24%, 0.43%, and 2.32% mAP. Using KLD loss also improves the detection of other vehicles such as large vehicles, and helicopters by 0.29% and 6.42% AP. Fig.10 shows the improvement in results of training S2A-Net-ILD with the KLD loss function instead of the smooth L1-loss.

2) COMPARISONS WITH THE STATE-OF-THE-ART

Table.4 shows a comparison of our model S2A-Net-ILD-KLD with state-of-the-art methods. S2A-Net-ILD-KLD achieves a competitive result of 80.39% overall mAP and It achieves 80.82% and 88.15% AP on small vehicles and storage tanks respectively. DODet [17] and Oriented R-CNN [26] are the two-stage models which outperform S2A-Net-ILD-KLD on the mAP metric. RPN for both DODet and Oriented R-CNN directly predicts oriented bounding boxes using horizontal anchor boxes for rotated RoI Align to extract features for the second stage. Therefore, they are less prone to overfitting than RoI-transformer and RRPN-based two-stage models, as they need a fewer number of trainable parameters. DODet also aligns the input features to its classifier using its regression output for RoI extraction. This slows the DODet model but it yields a better classification precision. Oriented R-CNN uses midpoint offset representation for its RPN output. This representation overcomes the issues from rotation regression by predicting offsets for the OBB vertices instead. R3Det with the KLD loss function (R3Det-KLD) outperforms S2A-Net-ILD-KLD only when using ResNet-152 as a backbone. When both R3Det-KLD and S2A-Net-ILD-KLD use ResNet-50, S2A-Net-ILD-KLD achieves a higher mAP. ResNet-152 is way slower and more memory-intensive than ResNet-50, so it isn't suitable for many setups and applications. It isn't feasible to train the model using ResNet-152 backbone on a single GPU without using a small batch size which isn't ideal.

V. CONCLUSION AND FUTURE WORK

In this paper, a new model has been proposed by incorporating Instance Level Denoising (ILD) into S2A-Net to improve predictions on small and cluttered objects such as small vehicles and ships. Our method improved the mAP value of S2A-Net to be 79.73% using the smooth L1-loss for regression on the Dota V1.0 dataset. S2A-Net with ILD has the downside of performance degradation on certain categories such as bridges. This stems from the ILD attention module adding more complexity to the model while there are fewer observations to train on some categories and not providing advantages over the baseline for larger objects. When the proposed model is trained using the Kullback–Leibler divergence (KLD) as a regression loss function, the model achieves a competitive result of 80.39% mAP, as the KLD loss function doesn't suffer from boundary discontinuity and a square-like problem, unlike the smooth L1-loss. This paper also compared the new model with the current state-of-the-art and specified reasons why the overall mAP of those models might be higher than our model. For future work to improve our model, a single anchor box is computationally efficient but it also increases the training difficulty as objects of different categories have different aspect ratios. Thus, adaptive anchor boxes or using different anchor boxes can improve results. Oriented R-CNN achieves very competitive results by using a midpoint offset representation for its OBB proposal predictions. This representation can be tested on single-stage models. Adding ILD to S2A-Net decreased the model inference speed from 16 FPS to 12.5 FPS which is around 21.875% slower. Thus, figuring out a different more lightweight attention module is worth investigating.

ACKNOWLEDGMENT

The authors would like to thank Prof. Ayman El-Dessouki for his support and his shared knowledge.

REFERENCES

[1] X. Ran, X. Zhou, M. Lei, W. Tepsan, and W. Deng, "A novel *K*-means clustering algorithm with a noise algorithm for capturing urban hotspots," *Appl. Sci.*, vol. 11, no. 23, p. 11202, Nov. 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/23/11202

[2] G. Cheng and J. Han, "A survey on object detection in optical remote sensing images," *ISPRS J. Photogramm. Remote Sens.*, vol. 117, pp. 11–28, Jul. 2016.

[3] J. Han, J. Ding, J. Li, and G.-S. Xia, "Align deep features for oriented object detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–11, 2021.

[4] X. Yang, J. Yan, X. Yang, J. Tang, W. Liao, and T. He, "SCRDet++: Detecting small, cluttered and rotated objects via instance-level feature denoising and rotation loss smoothing," *CoRR*, vol. abs/2004.13316, pp. 1–15, Apr. 2020.

[5] X. Yang, X. Yang, J. Yang, Q. Ming, W. Wang, Q. Tian, and J. Yan, "Learning high-precision bounding box for rotated object detection via Kullback–Leibler divergence," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1–14.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 21–37.

[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[10] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[11] Y. Yu, H. Guan, D. Li, T. Gu, E. Tang, and A. Li, "Orientation guided anchoring for geospatial object detection from remote sensing imagery," *ISPRS J. Photogramm. Remote Sens.*, vol. 160, pp. 67–82, Feb. 2020.

[12] C. Li, C. Xu, Z. Cui, D. Wang, Z. Jie, T. Zhang, and J. Yang, "Learning object-wise semantic representation for detection in remote sensing imagery," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Jun. 2019, pp. 20–27.

[13] X. Yang, H. Sun, X. Sun, M. Yan, Z. Guo, and K. Fu, "Position detection and direction prediction for arbitrary-oriented ships via multitask rotation region convolutional neural network," *IEEE Access*, vol. 6, pp. 50839–50849, 2018.

[14] J. Ding, N. Xue, Y. Long, G.-S. Xia, and Q. Lu, "Learning RoI transformer for oriented object detection in aerial images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2844–2853.

[15] C. Chen, W. Gong, Y. Chen, and W. Li, "Object detection in remote sensing images based on a scene-contextual feature pyramid network," *Remote Sens.*, vol. 11, no. 3, p. 339, Feb. 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/3/339

[16] X. Yang, H. Sun, K. Fu, J. Yang, X. Sun, M. Yan, and Z. Guo, "Automatic ship detection in remote sensing images from Google Earth of complex scenes based on multiscale rotation dense feature pyramid networks," *Remote Sens.*, vol. 10, no. 1, p. 132, Jan. 2018. [Online]. Available: https://www.mdpi.com/2072-4292/10/1/132

[17] X. Yang, J. Yan, Z. Feng, and T. He, "R3Det: Refined single-stage detector with feature refinement for rotating object," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 4, pp. 3163–3171. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16426

[18] X. Yang, J. Yan, Q. Ming, W. Wang, X. Zhang, and Q. Tian, "Rethinking rotated object detection with Gaussian Wasserstein distance loss," in *Proc. ICML*, 2021, pp. 11830–11841.

[19] J. Han, J. Ding, N. Xue, and G.-S. Xia, "ReDet: A rotation-equivariant detector for aerial object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2786–2795.

[20] W. Guo, W. Yang, H. Zhang, and G. Hua, "Geospatial object detection in high resolution satellite images based on multi-scale convolutional neural network," *Remote Sens.*, vol. 10, no. 1, p. 131, Jan. 2018.

[21] J. Ding, N. Xue, G.-S. Xia, X. Bai, W. Yang, M. Y. Yang, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "Object detection in aerial images: A large-scale benchmark and challenges," *CoRR*, vol. abs/2102.12219, 2021. [Online]. Available: https://arxiv.org/abs/2102.12219

[22] Z. Wei, D. Liang, D. Zhang, L. Zhang, Q. Geng, M. Wei, and H. Zhou, "Learning calibrated-guidance for object detection in aerial images," 2021, *arXiv:2103.11399*.

[23] X. Yang and J. Yan, "Arbitrary-oriented object detection with circular smooth label," in *Computer Vision—ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham, Switzerland: Springer, 2020, pp. 677–694.

[24] X. Yang, L. Hou, Y. Zhou, W. Wang, and J. Yan, "Dense label encoding for boundary discontinuity free rotation detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15819–15829.

[25] J. Wang, W. Yang, H.-C. Li, H. Zhang, and G.-S. Xia, "Learning center probability map for detecting objects in aerial images," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4307–4323, May 2021.

[26] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3520–3529.

[27] X. Zheng, W. Zhang, L. Huan, J. Gong, and H. Zhang, "AProNet: Detecting objects with precise orientation from aerial images," *ISPRS J. Photogramm. Remote Sens.*, vol. 181, pp. 99–112, Nov. 2021.

[28] G. Cheng, Y. Yao, S. Li, K. Li, X. Xie, J. Wang, X. Yao, and J. Han, "Dual-aligned oriented detector," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–11, 2022.

[29] X. Yang, J. Yang, J. Yan, Y. Zhang, T. Zhang, Z. Guo, X. Sun, and K. Fu, "SCRDet: Towards more robust detection for small, cluttered and rotated objects," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8231–8240.

[30] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 801–818.

[31] G. Zhang, S. Lu, and W. Zhang, "CAD-Net: A context-aware detection network for objects in remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 10015–10024, Dec. 2019.

[32] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, "R2 CNN: Rotational region CNN for arbitrarily-oriented scene text detection," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 3610–3615.

[33] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3111–3122, Nov. 2018.

[34] J. Huang, V. Sivakumar, M. Mnatsakanyan, and G. Pang, "Improving rotated text detection with rotation region proposal networks," 2018, *arXiv:1811.07031*.

[35] Y. Ya, H. Pan, Z. Jing, X. Ren, and L. Qiao, "Fusion object detection of satellite imagery with arbitrary-oriented region convolutional neural network," *Aerosp. Syst.*, vol. 2, no. 2, pp. 163–174, Nov. 2019.

[36] M. Weiler and G. Cesa, "General E(2)-equivariant steerable CNNs," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2019, pp. 1–12.

[37] J. Wang, J. Ding, H. Guo, W. Cheng, T. Pan, and W. Yang, "Mask OBB: A semantic attention-based mask oriented bounding box representation for multi-category object detection in aerial images," *Remote Sens.*, vol. 11, no. 24, p. 2930, Dec. 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/24/2930

[38] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11482–11491.

[39] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. CVPR*, Jun. 2016, pp. 3150–3158.

[40] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2016, pp. 1–13.

[41] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.

[42] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, "Oriented response networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4961–4970.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, F. Bach and D. Blei, Eds. Lille, France, Jul. 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[45] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*. Madison, WI, USA: Omnipress, 2010, pp. 807–814.

[46] Y. Xue. (Dec. 2020). *DOTA-DoAI—Github Repository*. [Online]. Available: https://github.com/SJTU-Thinklab-Det/DOTA-DOAI

[47] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3974–3983.

[48] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[49] K. Chen *et al.*, "MMDetection: Open MMLab detection toolbox and benchmark," 2019, *arXiv:1906.07155*.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 2015, pp. 1–15.

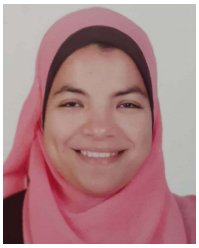
[51] T. Tieleman and G. Hinton, "Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude," in *Neural Networks for Machine Learning*. Mountain View, CA, USA: COURSERA, 2012.

[52] J. Han. (2021). *S2ANet*. [Online]. Available: https://github.com/csuhan/s2anet

YASSIN ZAKARIA was born in Cairo, Egypt, in 1994. He received the B.S. degree in computer engineering from the Faculty of Engineering, Cairo University, in 2017. He began working as a Research Assistant with the Department of Computers and Systems, Electronics Research Institute (ERI), in 2018. His research interests include computer vision and machine learning.

SAHAR A. MOKHTAR received the B.Sc., M.Sc., and Ph.D. degrees from the Electronics and Communications Department, Faculty of Engineering, Cairo University, in 1990, 1994, and 2004, respectively. She is a Researcher with the Electronics Research Institute (ERI). She joined ERI, in 1991. She is also a part-time Lecturer for AI, data mining, and database courses at many private universities, such as the Misr University for Science and Technology and the Akhbar Elyoum Academy. Her research interests include artificial intelligence, machine learning, and data and web mining.

HODA BARAKA received the B.Sc., M.Sc., and Ph.D. degrees from the Electronics and Communications Department, Faculty of Engineering, Cairo University, in 1982, 1985, and 1989, respectively. She was the Former Head of the Computer Engineering Department, Faculty of Engineering, Cairo University, from 2019 to 2020, where she is currently a Professor, and an Advisor to the Minister for Technology Talents Development. She is an experienced Information Communication Technology Advisor with a demonstrated history of working in the government administration industry. She is skilled in enterprise architecture, ITIL, strategy, IT strategy, and public policy. Her research interests include computer networks, blockchain, information systems, the Internet of Things, and artificial intelligence.



MAYADA HADHOUD received the B.Sc., M.Sc., and Ph.D. degrees from the Computer Engineering Department, Faculty of Engineering, Cairo University, in 2004, 2010, and 2014, respectively. She is an Assistant Professor with the Computer Engineering Department, Faculty of Engineering, Cairo University. Her research interests include image processing, computer vision, and machine learning.