VEHICLE INSTANCE SEGMENTATION WITH ROTATED BOUNDING BOXES IN UAV IMAGES USING CNN

S. El Amrani Abouelassad*, F. Rottensteiner

Institute of Photogrammetry and GeoInformation, Leibniz Universität Hannover, Germany -
(elamrani, rottensteiner)@ipi.uni-hannover.de

Commission I, WG I/2

KEY WORDS: Remote Sensing, Vehicle Detection, Mask RCNN, Deep Learning, Instance Segmentation.

ABSTRACT:

Vehicle instance segmentation is a major but challenging task in aerial remote sensing applications. More importantly, the current majority methods use horizontal bounding boxes which does not tell much about the orientation of vehicles and often leads to inaccurate mask proposals due to high background to foreground pixel-ratio. Given that the orientation of vehicles is important for numerous applications like vehicle tracking, we introduce in this paper a deep neural network to detect and segment vehicles using rotated bounding boxes in aerial images. Our method demonstrates that rotated instance segmentation improves the mask predictions, especially when objects are not axis aligned or are touching. We evaluate our model on the ISPRS benchmark dataset and our newly introduced UAV dataset for vehicle segmentation and show that we can significantly improve the mask accuracy compared to instance segmentation using axis-aligned bounding boxes.

1. INTRODUCTION

There has been growing interest in the use of unmanned aerial vehicles (UAVs) in engineering applications such as environmental monitoring, surveillance, and traffic monitoring, to name a few. The availability of this solution has been beneficial for various applications, because UAVs allow for an easy and fast acquisition of detailed images of an area of interest with resolutions tailored to the task. The application of interest in this paper is traffic surveillance at critical points, e.g. dangerous cross-roads. For such an application, UAVs could be considered as a part of an infrastructure that can monitor that dangerous spot. This might become even more relevant in the context of autonomous driving, considering that autonomous cars are expected to exchange information with other cars or with some relevant objects of infrastructure, a scenario that has also been investigated in (Schön et al., 2018).

In this setup, the goal of processing the images produced by an UAV is the detection and tracking of all the vehicles in the field of view of its camera. This paper focuses on the first step, the detection of vehicles in high-resolution near-nadir-view images acquired by an UAV using a Convolutional Neural Network (CNN). Classical methods of object detection deliver a bounding box aligned with the image rows and columns for every instance of an object in the scene (Pinheiro et al., 2015); methods for instance segmentation additionally deliver a binary segmentation mask for every instance, indicating the set of pixels inside the bounding box that belong to the object (He et al., 2017).

Nevertheless, in near-nadir images, vehicles can be distributed in arbitrary orientations, sometimes being densely packed in areas of of high traffic. In such cases, axis-aligned bounding boxes will usually contain other neighbouring objects and a lot of background information, so using such boxes would not seem to be a natural choice for our problem. Vehicles typically have an almost rectangular footprint in near-nadir imagery, and it would be a much more obvious choice to predict oriented bounding boxes, i.e. rotated rectangles whose main orientation is identical to the orientation of the car, in the detection step. Such a representation of a vehicle by a rotated would already represent the shape of the vehicle very well; however, the binary mask which is predicted in instance segmentation would give an even more detailed representation of the shape of the car nevertheless. If the representation of a car at instance level is given by a rotated bounding box, we believe it is also preferable to align the instance mask to be predicted to that rotated bounding box, thus avoiding boxes with many background pixels. The information about the rotation of the bounding boxes could be extracted from the results of instance segmentation in a post-processing step, e.g. by determining the minimum bounding rectangle (MBR) from the binary mask. In this paper, we pursue another strategy: the goal of this paper is to directly determine oriented bounding boxes and by a CNN and a binary mask aligned with the rotated box. Knowledge about the orientation of the vehicle will also be an important cue for tracking, which will be tackled in future work. We want to investigate whether using oriented bounding boxes would help in better performance and results for detection and segmentation of vehicles.

There are methods for predicting rotated bounding boxes in object detection, e.g. (Liu et al., 2017) or (Jiang et al., 2017), but instance segmentation was not carried out, and region proposal is not based on a CNN. Mou and Zhu (2018) proposed a semantic boundary-aware Res-FCN (Fully Convolutional Network) for vehicle instance segmentation that does not use axis-aligned bounding boxes; in fact, it does not use bounding boxes at all, predicting object boundaries along with the instance masks instead. Given the fact that vehicles can be represented by rectangles quite well in near-nadir views, this would seem to be a rather complex approach.

Our method detects vehicles in near-nadir images, e.g. acquired from a UAV, and predicts a rotated bounding box for

* Corresponding author

every detected instance along with a binary mask identifying pixels belonging to the object. The method is based on Mask-RCNN, proposed for instance segmentation in (He et al., 2017). Our main methodological contribution is an extension of that method to predict rotated bounding boxes and binary instance masks aligned with the rotated boxes. This also requires a modification of the training procedure. We also introduce a new dataset consisting of high-resolution images acquired using a UAV and a reference for instance segmentation with rotated bounding boxes. Using this new dataset and an existing benchmark we evaluate the performance of bounding box prediction and instance segmentation and show that it can achieve good results.

2. RELATED WORK

2.1 Object Detection

The goal of object detection is to detect all instances of objects belonging to some pre-defined classes of interest and to localize them in the image, usually represented as bounding boxes with confidence scores for the classes of interest and the background. For vehicle detection in aerial images, early works relied on the use of hand crafted visual features and classifiers. Shao et al. (2012) detect vehicles from high resolution aerial images using visual features such as histograms of oriented gradients (HOG) (Dalal and Triggs, 2005) and local binary pattern (Ojala et al., 2002). Moranduzzo and Melgani (2014) detect vehicles by firstly using SIFT (Lowe, 1999) to detect vehicle interest points and training a support vector machine to classify the detected interest points according to whether they correspond to a vehicle or not. However, these methods rely heavily on manual feature engineering.

The use of deep learning, especially of CNNs, has achieved amazing success in object detection. CNNs have a strong ability to learn image features (Zhao et al., 2019). Object detection based on CNN is typically performed in two stages. In the first stage, candidate regions that are likely to contain objects are detected. In the second stage, the image content inside each candidate box is used for classification and to improve the geometrical accuracy of the bounding box by regression, usually using two separate network branches, e.g. (Girshick, 2015; Ren et al., 2017). The classification output may also assign a box to the background class, in which case the candidate is discarded; otherwise, the class label indicates the object type. Early work used separate region proposal methods not based on CNN (Girshick et al., 2014). This is also true for Fast R-CNN (Girshick, 2015), where a backbone CNN was incorporated to produces a feature map for the entire image, from which the features inside of the bounding boxes were extracted by a Region of Interest (RoI) pooling layer. These features are processed by a CNN with two branches, one for classification and one for bounding box regression. Faster R-CNN (Ren et al., 2017) additionally proposed a region proposal network (RPN), i.e. a CNN that uses the feature map generated by a backbone network to predict candidate bounding boxes for Fast R-CNN. The RPN uses a series of axis-aligned anchor boxes of different size and aspect ratio; each pixel in the input image is considered to be a potential centre of a bounding box corresponding to each of the anchors, and the RPN predicts an objectness score and regresses the bounding box parameters for all of these potential bounding boxes. After non-maxima suppression, the N proposals having the best objectness scores are selected for further processing. Feature Pyramid Networks (FPN) (Lin et al., 2017) generating multi-scale feature maps to locate objects, to improve the object detection results for cases in which objects may appear at very different scale. However, using a backbone with less semantic information affects the generated multi-scale features because they will not be fully exploited. All the methods mentioned such use axis-aligned bounding boxes. However, for applications like vehicle monitoring and tracking, the orientation of the detected vehicles is a crucial piece of information. Yet, work on detection of vehicles that also predicts the vehicle orientation, which would be performed implicitly if rotated bounding boxes were considered, has found less attention, and predicting the orientation is still considered as a challenge.

2.2 Detection with Rotated Bounding Boxes

There are different strategies for orientation estimation of objects. A classical method is presented in (Liu and Mattyus, 2015), where a fast binary detector using integral channel features in a soft-cascade structure is used to detect vehicles. Afterwards, HOG features are used to classify the orientations of the detected vehicles. One of the disadvantages of this method is that it is based on hand-crafted features. Deep learning methods for oriented box detection have also been proposed. Han et al. (2021) propose a rotation-equivariant detector for aerial object detection that uses a backbone neural network which encodes rotation equivariance and rotation invariance to extract rotation-equivariant features, and propose rotation-invariant RoI Align to extract rotation-invariant features from rotation-equivariant features. Yi et al. (2021) describe oriented objects in the same Cartesian coordinate system as boundary-aware vectors and use an extended horizontal keypoint-based object detector to regress them. Yang et al. (2019) detect rotated objects with a single-stage detector using a progressive regression approach. However, the exact orientation of the objects in $(0, 2\pi]$ was not given much importance in the mentioned deep learning methods.

Jiang et al. (2017) use an RPN based on axis-aligned anchors for inferring axis aligned bounding boxes before regressing the orientation. Xia et al. (2018) suggest to extend state-of-the-art methods for object detection to regress a rotated bounding box for objects of interest, and they also propose a benchmark dataset (DOTA) for oriented object detection. Nonetheless, they use axis-aligned anchors for the training of the RPN (instead of rotated anchors used in our work), and they regress the coordinates of the bounding box (i.e, the 8 coordinates of the 4 positions of the bounding box vertices in the image). Thus, the bounding box regression is over-parameterized, which may be problematic and result in rotated boxes that are not rectangular. Ding et al. (2019) also introduces an approach for generating rotated Region of Interests (RoI Transformer). It is based on the output of axis-aligned RPN. First, the RoI Transformer learns to transform an axis-aligned region of interest into a rotated region of interest. Afterwards, a second module extracts rotation-invariant features from the rotated RoIs to be used for classification and regression. The authors claim that using rotated anchors would result in too many anchors to be considered; this may be true for a method designed for different target objects with various sizes and aspect ratios. However, as we restrict ourselves to a single object category (vehicles), we can restrict the number of aspect ratios considered based on object knowledge, so that it becomes feasible to use rotated anchor boxes. A CNN for ship detection, representing ships by rotated bounding boxes, was introduced by Liu et al. (2017). Using rotated region proposals extracted using the method presented in (Liu et al., 2016), a rotated region of interest pooling layer is applied before regressing the bounding boxes. However, the region proposal method is not based on a CNN.

2.3 Instance Segmentation

Many instance segmentation methods are based on Faster R-CNN (Ren et al., 2017), expanding the network by a branch for predicting a binary mask identifying pixels inside the bounding boxes that correspond to the object of interest. An example is Mask R-CNN (He et al., 2017). The architecture of the mask prediction branch is similar to the decoder of a fully convolutional network for semantic segmentation (Long et al., 2015). Mask Scoring R-CNN (Huang et al., 2019) learns to determine the quality of the mask predictions, adapting the predicted scores of the masks in a way that high scores are given to high quality masks and lower scores are given to the lower quality masks. YOLACT (Bolya et al., 2019) enhances the speed of instance segmentation by using a linear combination of prototypes for mask prediction and a fast version of non-maximum-suppression. Shape priors are also used in (Kuo et al., 2019), mainly with the goal to improve the generalization performance of the model and to reduce the amount of training data.

However, all of the methods mentioned so far use axis-aligned bounding boxes for instance segmentation. Mou and Zhu (2018) proposed a semantic boundary-aware Res-FCN (Fully Convolutional Network) to compute vehicle instance segmentation by predicting vehicle regions and their boundaries, not considering rectangular bounding boxes at all. The network residual blocks feature representations are upsampled in two separate but identical branches to segment vehicles and their boundaries. However, the heading of vehicle instances could only be determined from the results in post-processing. In this paper, we try to address the vehicle instance segmentation problem by a end-to-end learning framework using rotated bounding boxes and binary object masks aligned with these rotated boxes.

3. METHODOLOGY

The goal of our method is to extract the instances of vehicles from an aerial near-nadir view image. It is based on Mask-RCNN (He et al., 2017), adapting it to predict a rotated bounding box and a binary instance mask aligned with that box for every detected instance and using an end-to-end training framework. Figure 1 shows an overview of the network used in this paper. The network uses a ResNet50 (He et al., 2016) backbone to extract features from the image. Afterwards, a new rotated region proposal network (*RRPN*) is used to extract a limited number of RoIs that are to considered candidate regions for containing vehicles. Similarly to the RPN in (He et al., 2017), the RRPN is based on a set anchor boxes, considering every pixel in the feature map generated by the backbone to be a potential centre of one RoI per anchor, the shape of each of these candidates being defined by the anchor. The RRPN also regresses box parameter updates relative to the anchor for every candidate and selects the best candidates according to a classification score. For each candidate object thus extracted by the RRPN, a rotated RoI (*RRoI*) pooling layer extracts a feature map of a fixed size aligned with the rotated bounding box of the candidate. As in (He et al., 2017), this feature map is processed by two network branches. The first one is a classification and regression head that predicts whether a candidate corresponds to a vehicle or not and improves the parameters of the bounding box; the second one predicts a binary instance mask aligned with the bounding box, which identifies pixels inside the box that correspond to the vehicle.

The main modifications of our method compared to (He et al., 2017) are the RRPN and the RRoI layer. Unlike the RPN (He et al., 2017), the RRPN extracts RoI that are rotated with respect to the image coordinate system. This is achieved by using a set of anchors that represent rectangles of different orientations and sizes in the RRPN. As the only object type we are interested in are vehicles, we exploit the knowledge about the shape of vehicles to limit the number of used anchors with respect to scales and aspect ratios. As a consequence of allowing for rotated bounding boxes, RoI regression additionally has to predict the orientation angle of the rectangle. This information is used in the RRoI pooling layer to align the extracted features with the rotated bounding boxes. Of course, the final bounding box regression branch also has to refine five bounding box parameters rather than four, as done in (He et al., 2017). The rotated RoI features are fed the FCN branch, which predicts the binary map of vehicle pixels inside the rotated bounding box, which will also be aligned with the rotated box. In the following subsections, we give more details about the main parts of our framework and the training procedure.

3.1 Rotated Bounding Box Representation

As shown in Figure 2, every bounding box is represented by five parameters, collected in a parameter vector $\mathbf{b} = (r, c, l_1, l_2, \theta)$. It contains the image coordinates of the vehicle centre (r, c), the angle $\theta \in (0, 2\pi]$ representing the orientation of the main vehicle axis relative to the x axis of the image coordinate system, and the lengths l_1 and l_2 of the longer and the shorter semi-axes of the oriented rectangular box, respectively. In general, the direction of the main vehicle axis is defined to be the driving direction. In our experiments, we will also test our method using data in which the driving direction of the vehicles is not given in the reference data. In these cases, we restrict the rotation angles to the interval $(0,\pi]$, defining the main direction of the vehicle to correspond to the longer semi-axis of the rectangle.

3.2 Rotated Region Proposal Network (RRPN)

Similarly to the RPN proposed in (Ren et al., 2017), the RRPN uses a window of size 3×3 window that slides over the feature map generated by the backbone. The features inside the window are resampled to a 256 dimensional vector, which is passed to two fully connected layers. The first of these layers computes a confidence score indicating whether the centre of the 3×3 corresponds to an object described by a certain anchor or not, and the second one regresses offsets of the five parameters of the bounding box with respect to the parameters derived from the corresponding anchor. As we want to predict oriented boxes, we use oriented anchors with different orientations and sizes to feed the RRPN. Ding et al. (2019) claims that this results in a higher number of anchors compared to axis-aligned boxes. However, we are interested in vehicles only, so that we can exploit the knowledge about the typical shape of vehicles to limit the number of used anchors with respect to scale and aspect ratio. Unlike (He et al., 2017), we only use one aspect ratio ($l_1 : l_2 = 2 : 1$), a small number of scales (1 or 2, depending on the dataset), but several rotations (11 or 6, depending on whether we have a reference for the driving direction or not).

The classification layer generates two class scores (one for *object* and one for *no object*) using the softmax function, and the

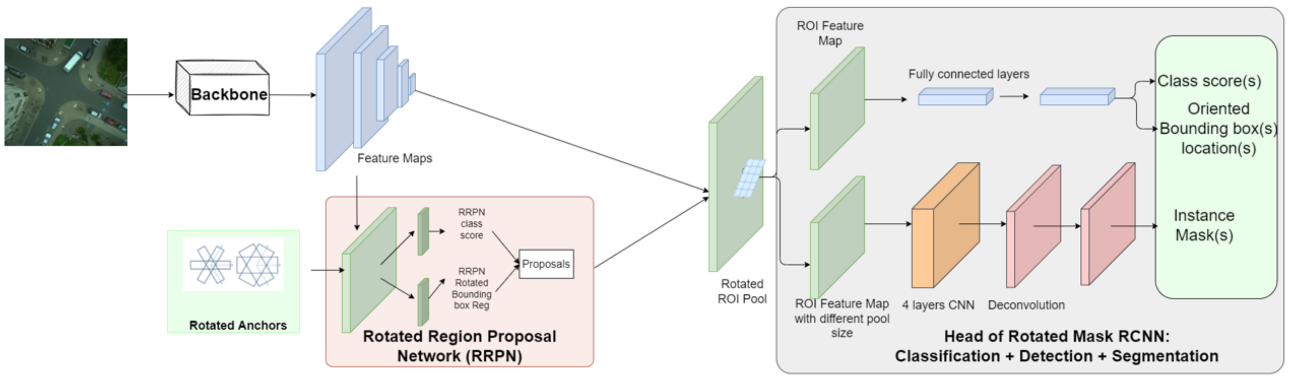
Backbone
Feature Maps
Rotated Anchors
RRPN class score
RRPN Rotated Bounding box Reg
Proposals
Rotated Region Proposal Network (RRPN)
Rotated ROI Pool
ROI Feature Map
Fully connected layers
Class score(s)
Oriented Bounding box(s) location(s)
Instance Mask(s)
ROI Feature Map with different pool size
4 layers CNN
Deconvolution
Head of Rotated Mask RCNN: Classification + Detection + Segmentation

Figure 1. The architecture of the proposed method.

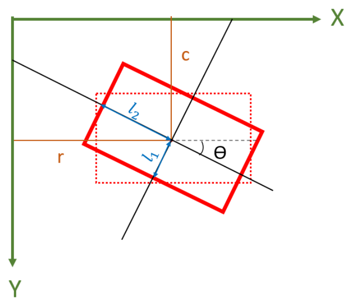

Figure 2. Rotated Bounding Box representation.

box regression layer delivers a correction to each of the five parameters per position and anchor. Similarly to (Ren et al., 2017), cross-boundary predictions are removed in the training phase, but only clipped to the image boundaries at inference time. Then, non-maximum suppression is used to reduce the number of proposed RRoIs. For every initial region proposal, we check its overlap with other region proposals. We determine the intersection over union (IoU) score of each overlapping pair of candidate boxes; if a candidate has an IoU score larger than 0.75 with another box that has a higher classification score, it is discarded. The remaining candidates are ranked according to their scores for the foreground class, and the N_{prop} (set to $N_{prop} = 1000$ in our experiments) best-ranked candidates are selected as the proposed regions that are forwarded to the RRoI pooling layer.

3.3 Rotated RoI Pooling

For every region proposals generated by the RRPN, a feature map is extracted in the RRoI pooling layer. This feature map has to be aligned with the proposed bounding box. The size of the feature map is $m \times m$ (we use $m = 14$ in our experiments). The bounding box is divided into a a grid of $m \times m$ rectangular bins the sides of which are parallel to the edges of the bounding box. Using the feature map produced by the ResNet backbone as input, for each of these bins we identify all input feature vectors inside the rectangle boundaries, and one representative feature vector is generated per bin using max-pooling among all of these feature vectors. The output feature map corresponding to the RRoI is transferred to the classification, box regression and mask prediction heads of the CNN.

3.4 Classification and Bounding Box Regression

For the classification of vehicles, the fixed-sized feature map from RoI Pooling is processed by a sequence of fully connected layers that are identical to the ones used in (Ren et al., 2017), except for the number of nodes in the last layers of the output branches. There are two such output branches. The first one generates softmax scores for the classes *vehicle* and *no vehicle*, whereas the second one predicts 5 real-valued numbers encoding the bounding parameters.

3.5 Mask Prediction

The feature maps produced by the RRoI layer, having a spatial dimension of $m \times m = 14 \times 14$, are fed into a sequence of 4 convolutional layers with rectified linear unit activation function (ReLU), followed by an upsampling layer with a factor 2, using transposed convolution. Finally, a sigmoid activation is applied to every pixel of the resultant feature map (spatial dimensions: 28×28) to get the score for the pixel to belong to the foreground. Similar to (He et al., 2017), an affine transformation is applied to the map of class scores thus generated to get a new map that has square pixels at the geometrical resolution of the input images, using bilinear resampling in this process. The final binary mask of vehicle pixels is obtained by binarizing the resampled score map using a threshold of 0.5.

3.6 Training

The training data consist of images with known rotated bounding boxes enclosing vehicles and binary masks identifying vehicle pixels inside the bounding boxes. We use a stratiefied training procedure similarly to (Ren et al., 2017) and (He et al., 2017), but using adapted versions of the loss functions. The ResNet backbone is pre-trained on ImageNet (Deng et al., 2009). The parameters of the RRPN and the classification and box regression branches are initialized by the stratified procedure of (Ren et al., 2017), in which one component of the CNN is trained after the other, taking into account different components of the loss function described below. After that, the results of the RRPN are used to train the instance segmentation branch of the network. Finally, similarly to (He et al., 2017), we fine-tune the network using a combined loss function considering all intermediate and final outputs, which is the reason why we speak about end-to-end training. The multi-task loss L_{total} to be minimized in training consist of four terms:

$$L_{total} = L_{RRPN} + L_{cls} + L_{reg} + L_{mask}. \qquad (1)$$

The four loss terms will be explained in the subsequent subsections.

3.6.1 RRPN Loss L_{RRPN}:

This loss is computed from the output of the RRPN. Each proposed anchor is classified according to whether it corresponds to an object bounding box or not, and the parameters of the bounding box are regressed. For each such anchor, we have to define whether it corresponds to an object, i.e. to a sample for class foreground, or not, i.e. to a sample for class background. As in (Ren et al., 2017), this is decided according to its IoU score with the ground truth bounding boxes. An anchor is assigned to the foreground if its IoU with a ground truth box is higher than fg_{Thresh} = 0.7. If multiple anchors have such a large overlap with a ground truth box, only the one having the highest IoU is maintained and the others are discarded. On the other hand, an anchor is assigned to the background if its IoU with any ground truth box is lower than 0.3. After removing candidates crossing the image boundaries and non-maxima suppression, about 2000 candidate boxes are extracted during training, similarly to (Ren et al., 2017), among which there will be both positive and negative samples. These 2000 boxes are further reduced to 256 (i.e. 128 per class) by random sampling.

The loss is based on this reduced set of samples for an input image. It consists of two terms:

$$L_{RRPN} = L_{RRPN_{cls}} + L_{RRPN_{reg}}. \tag{2}$$

The first term, L_{RRPN_cls}, is computed at the end of the classification branch of the network (*object* vs. *no object*). Similarly to the classification loss L_{cls} in (Ren et al., 2017), we use a log loss for the binary classification task defined as:

$$L_{RRPN_{cls}} = -\frac{1}{N}\sum_{i=1}^{N} l_i \cdot log(p_o^i) + (1-l) \cdot log(p_{no}^i), \tag{3}$$

where l_i is the label of the sample i (1 for *object* anchors and 0 for *no object* anchors), p_o^i and p_{no}^i are the softmax outputs for the *object* and *no object* classes, respectively, for that sample, and N is the number of samples (256 if one image is processed at a time in training).

The second term in eq. 2, $L_{RRPN_{reg}}$ measures the level of agreement of the bounding box parameters with the reference in a way similar to the regression loss in (Ren et al., 2017). It considers the differences of the known and the predicted offsets of the bounding box compared to the anchor. These offsets t_p with $p \in \{r, c, l_1, l_2, \theta\}$ are defined as follows :

$$\begin{aligned} t_c &= \frac{(c - c_a)}{l_{1a}} \\ t_r &= \frac{(r - r_a)}{l_{2a}} \\ t_{l_1} &= log(l_1) - log(l_{1a}) \\ t_{l_2} &= log(l_2) - log(l_{2a}) \\ t_\theta &= \theta - \theta_a + s2\pi, \end{aligned} \tag{4}$$

where $(c_a, r_a, l_{1a}, l_{2a}, \theta_a)$ are the parameters of the box corresponding to the anchor. Analogously, an offsets t_p^* for every parameter of a reference bounding box is defined. Taking these definitions, for every parameter $p \in \{r, c, l_1, l_2, \theta\}$, there is one loss term measuring the difference $\Delta p = t_p^* - t_p$ between the true and the predicted offsets. The parameter $s \in Z$ is used to ensure that $t_\theta \in (0, 2\pi]$. Similarly to the RPN regression loss of (Ren et al., 2017), we use the loss

$$L_{RRPN_reg} = \frac{1}{N_{reg}} \cdot \sum_{n=1}^{N_{reg}} \sum_{p\in\{r,c,l_1,l_2,\theta\}} L_H(\Delta p_n), \tag{5}$$

where Δp_n is the difference of the offsets of n^{th} bounding box and L_H is the Huber loss function, referred to as smooth L1 loss in (Girshick, 2015):

$$L_H(\Delta p) = \begin{cases} 0.5\Delta p^2 & \text{if } |\Delta p| < 1 \\ |\Delta p| - 0.5 & \text{otherwise} \end{cases} \tag{6}$$

The sum in eq. 5 is taken over the N_{reg} samples for the *object* class. The main difference of this loss and the corresponding one in (Ren et al., 2017) is that it includes a term for the offset of the rotation angle of the bounding box.

3.6.2 Classification Loss L_{cls}:

This loss is computed at the end of the classification branch of the network. Similarly to (Girshick, 2015), we use a log loss for the binary classification task (*vehicle* vs. *no vehicle*):

$$L_{cls} = -\frac{1}{N}\sum_{i=1}^{N} log(p^i(C^i)), \tag{7}$$

where $p^i(C^i)$ denotes the softmax output for the correct class label C^i of the i^{th} candidate box and N is the number of such boxes.

3.6.3 Regression Loss L_{reg}:

This loss is computed at the end of the bounding box regression branch of the network. It is similar to L_{RRPN_reg} in eq. 5:

$$L_{reg} = \frac{1}{N_{reg}} \cdot \sum_{n=1}^{N_{reg}} \sum_{p\in\{r,c,l_1,l_2,\theta\}} L_H(\Delta_{p'_n}). \tag{8}$$

In eq. 8, all terms are identical to those of eq. 5, except of the argument of the loss function L_H. Here we use the differences $\Delta'_p = p^* - p$ of the predicted parameter p and the ground truth value p^*, e.g. $\Delta'_r = r^* - r$ for the parameter r.

3.6.4 Mask Loss L_{mask}:

This loss is computed at the end of the branch for predicting the binary object masks. This branch generates a mask of dimension $o \times o$ for each RRoI (we use o=28). Similarly to (He et al., 2017), we use the binary cross-entropy loss for every pixel of the upsampled feature maps for the rotated bounding box k:

$$\begin{aligned} L_{mask} = -\frac{1}{O}\sum_{k=1}^{N_{obj}} \sum_{1\le i,j\le o} [l_{ij}^k \cdot log(p_{ij}^k) + \\ +(1-l_{ij}^k) \cdot log(1-p_{ij}^k)], \end{aligned} \tag{9}$$

where $O = o^2 \cdot N_{obj}$ is the total number of terms in the sum, N_{obj} is the number of *object* bounding boxes considered, (i, j) identifies a cell in the feature maps, l_{ij}^k is the true binary label of that cell (1 for *object* and 0 for *no object*) for sample k, and p_{ij}^k is the corresponding sigmoid output for that pixel to correspond to the foreground *object*.

4. EXPERIMENTS AND RESULTS

4.1 Test Datasets

In our experiments, we use two datasets: the Potsdam dataset from the ISPRS Semantic Labeling challenge (Wegner et al., 2017) and a new dataset acquired by UAV specifically for this project. The ISPRS dataset consists of 24 tiles (6000×6000 pixels each) with 5 cm spatial resolution and a reference for pixel-wise classification of land cover. The reference data required by our method, in particular the rotated bounding box for each car instance, were generated from the pixel-based annotations for the class *car* provided by that benchmark. First, connected components of car pixels were extracted. Afterwards, a minimum bounding rectangle was determined for each of these components to derive the rotated bounding box, defining the rotation of the box to correspond to the longer semi-axis of the rectangle in the interval $(0, \pi]$; this was done because the driving direction of a car cannot be inferred from a label image automatically. The results of this process were checked interactively and corrected if required. Of course, the set of car pixels forming the basis of the generation of the rotated bounding box formed the reference for the binary instance masks. Altogether, there were 31564 instances of cars in this dataset. For all datasets used in the evaluation, we split the images into tiles of 256×256 pixels.

The second dataset, to which we refer as the UAV dataset in this paper, was generated following a measurement campaign conducted by us. It covers the scenario of a UAV with a camera hovering over a street intersection for the purpose of detecting and tracking vehicles. It contains parked vehicles and vehicles driving in different directions. In total, there are 3200 images each with a size of 2592×2048 pixels and 3 cm spatial resolution, acquired at a frequency of 10 Hz from the hovering UAV. The data were manually annotated. We differentiate two subsets of this dataset. For the first subset, consisting of 2600 images, binary masks of vehicle pixels were digitized manually, and the reference bounding boxes were generated in a way similar to the ISPRS dataset, so that the rotation is defined to be in the interval $(0,\pi]$. The number of car instances in this dataset was 23429, and we refer to it as dataset UAV_{180}. For the second subset, consisting of the remaining 600 images, the rotation of the bounding box was interactively corrected so that it refers to the driving direction of vehicle. Thus, for this subset, which consists of 4435 car instances, the angles are given in the interval $(0,2\pi]$. We refer to it as dataset UAV_{360}.

4.2 Test Setup

Training was based on the method described in section 3.6, using stochastic gradient descent with momentum and weight decay for optimization. The weight decay and momentum parameters were set to 0.0001 and 0.9, respectively. As a backbone we used ResNet-50 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009). The stratified training procedure was carried out in a way similar to (He et al., 2017). In the final training stage, in which the joint loss in eq. 1 was optimized, we trained the models for 50 epochs. The number of training iterations per epoch is identical to the number of image tiles, as in each iteration one image tile of 256×256 pixels is used to update the parameters. For the first 30 epochs, the learning rate was set to 0.001, afterwards it was decreased to 0.0001.

We applied augmentation to increase the number of samples in the UAV dataset, taking random crops and applying random scales in the range [0.9, 1.1] and a random rotations in $[0, 360^o)$. For the subset UAV_{180}, this resulted in a set with 32790 instances; for the subset UAV_{360}, the number of vehicle instances in the augmented dataset was 6532. Each dataset was divded into a training set consisting of 60% of data, 20% of the data were used for validation, and the remaining 20% constituted the test sets. In case of the UAV dataset, the split was applied by assigning the original images (before tiling) to respective subsets, so that tiles generated by data augmentation would be assigned to the corresponding set of original images.

We trained two variants of the CNN described in this paper based on the available data. The first variant, referred to as V_{180}, was trained using the training data from the ISPRS and the UAV_{180} datasets. Thus, in this variant, the rotations of the bounding boxes are in the interval $(0, \pi]$. We used 12 anchors in this variant, considering one aspect ratio (2:1), two scales (64 and 128 for l_1, resp.), and six rotation angles (0^o, 30^o, 60^o, 90^o, 120^o, 150^o). This variant will be evaluated on the test sets of the ISPRS and UAV_{180} datasets. The second variant, referred to as V_{360}, is trained to regress rotations in $(0, 2\pi]$. As the UAV_{360} dataset is quite small, we use a combination of all three training sets for training this model. In this case, we used eleven anchors, considering the same aspect ratio as in V_{180}, one scale only (64), and eleven rotation angles (0^o, 30^o, 60^o, 90^o, 120^o, 150^o, 180^o, 210^o, 240^o, 270^o, 300^o, 330^o). The results are evaluated using the test set of the UAV_{360} dataset. The experiments involving that second variant should show the capability of our method to predict correct orientations related to the driving direction of vehicles.

As a baseline method for instance segmentation, we use a variant of our method that uses axis-alinged bounding boxes. The resultant model, referred to as V_b, can be seen as a variant of Mask R-CNN with a restricted set of two anchors (related to an aspect ratio of 2:1 and two scales, identical to those used in V_{180}). In this case, no rotations were regressed, and the standard RoI pooling approach used in (He et al., 2017). The data used to train this model were identical to those used for training variant V_{180}.

In the evaluation, we analyse three aspects of the performance: object detection, segmentation of the instance masks, and prediction of the orientation of the bounding box. In order to measure detection performance, we compare the predicted bounding boxes to the reference bounding boxes. A predicted bounding box is considered to be a true positive (TP) if it has an IoU score with a reference box above 50%, otherwise it is considered to be a false positive (FP). Similarly, a false negative (FN) is a reference box having an IoU smaller than 50% with the predicted results. Based on the number of TP, FP and FN instances, the precision, i.e. the percentage of detected instances that correspond to an instance in the reference, and the recall, i.e. the percentage of reference instances that were detected, is determined; the F1 score, i.e. the harmonic mean of precision and recall, is also reported. Additionnally, the mean Average Precision (mAP) score is calculated by computing the area under the precision-recall curve which is computed by calculating the average value of precision and recall at every confidence threshold. Similar to (He et al., 2017), IoU thresholds varying from 50% to 95% by 5% step are used.

For assessing the quality of the predicted rotation angles, we compute the cumulated differences between the predicted and the reference angles and then determine a histogram of these differences. As far as the evaluation of the binary masks is concerned, we define a TP pixel to be a pixel corresponding to the vehicle in both, the reference and the predicted masks, and FP and FN pixels analogously to the definitions given for instances. In this case, all reference foreground pixels in FN detections are considered to be FN pixels, and all foreground pixels in masks predicted for FP detections are considered to be FP pixels. Based on these definitions, we also determine precision, recall, the F1 score and mAP percentage on the basis of all pixels in the test set.

4.3 Results and Discussion

As explained in the previous section, we trained and tested two variants of our method. The evaluation of the results of the two variants are described separately in the two subsequent sections.

4.3.1 Evaluation of the variant V_{180}: In this variant, the rotations of the bounding boxes are onyl given in the range between $(0, \pi]$. Table 1 presents the evaluation metrics for object detection achieved by this variant on the test sets from the ISPRS and the UAV$_{180}$ datasets described in section 4.1. The evaluation metrics of the binary masks predicted in instance segmentation are shown in table 2. In both cases, the metrics achieved by our method (V_{180}) are compared to the baseline (V_b), which does not consider rotated bounding boxes.

As far as the performance in object detection is concerned, our method V_{180} outperforms the baseline in when applied to the ISPRS dataset by a relatively large margin, e.g. 3.4% in the F1 score and 2% in mAP percentage. When applied to the UAV$_{180}$ test set, our method achieves a slightly larger precision and recall; the F1 score, which is a trade-off between the two values is also slightly better for our approach, though only by a small margin (0.9%), while the difference of the mAP percentages is larger for our approach by 2.9%. In general, we consider the detection performance to be relatively good.

Data	ISPRS				UAV$_{180}$			
Model	Prec.	Recall	F_1	mAP	Prec.	Recall	F_1	mAP
V_{180}	81.4	74.2	77.6	45.0	82.6	75.4	78.8	49.1
V_b	76.5	72.1	74.2	43.0	81.7	74.5	77.9	46.2

Table 1. Precision (Prec.), Recall, F_1 score and mAP for object detection on the two test datasets, achieved by the variant V_{180} of our method and the baseline V_b.

Similar observations can be made for the pixel-wise evaluation of the binary masks generated by our method and the baseline (cf. table 2). Again our method achieves better metrics for the ISPRS dataset, with a difference of 0.3% in the F1 score and 2.9% in the mAP percentage. It would seem that on that dataset, the binary masks predicted by our method are more precise compared to those of the baseline. For the UAV$_{180}$ dataset, the results are of a similar quality, with a large overall advantage of our method (2.5% in F1 and 4.9% in mAP). Again, the quality of the results is considered to be relatively good.

Data	ISPRS				UAV$_{180}$			
Model	Prec.	Recall	F_1	mAP	Prec.	Recall	F_1	mAP
V_{180}	81.8	71.1	76.1	44.1	82.4	77.8	80.0	50.0
V_b	79.8	72.3	75.8	41.2	81.3	74.2	77.5	45.1

Table 2. Precision (Prec.), Recall, F_1 score and mAP for the binary masks predicted on the two test datasets by the variant V_{180} of our method and the baseline V_b.

4.3.2 Evaluation of the variant V_{360}: In this variant, the rotations of the bounding boxes are given in the range between $(0, 2\pi]$, i.e. the predicted rotation refers to the driving direction of the vehicles. Figure 3 and Figure 4 show some qualitative results of V_{360} on UAV$_{360}$ test images. Table 3 presents the evaluation metrics for object detection achieved by this variant on the test sets from the UAV$_{360}$ datasets described in section 4.1. The evaluation metrics of the binary masks predicted in instance segmentation are shown in table 4. Again, the metrics achieved by our method (V_{360}) are compared to the baseline (V_b), which does not consider rotated bounding boxes.

In object detection, our method (V_{360}) performs better than the baseline by 0.7% in the F_1 score and 1% in mAP percentage. The evaluation of the predicted binary masks shows that our model (V_{360}) performs better compared to the axis-aligned model, with difference in F_1 score of about 2,6% and 1.4% in mAP percentage. In general, the performance on the UAV$_{360}$ dataset is slightly lower than the one of UAV$_{180}$ in terms of the detection quality, whereas it is a bit better in terms of a prediction of the binary masks. Both models V_{180} and V_{360} show better results in predicting vehicle instances compared with the baseline, and results demonstrates that models using rotated bounding boxes outperform models using axis-aligned boxes in the prediction of instance vehicle masks.

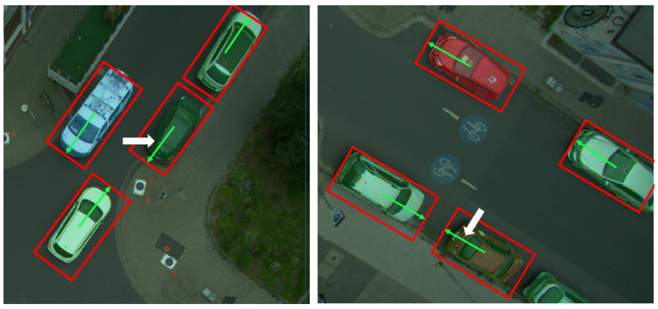
Figure 3. Qualitative results of V_{360} on UAV$_{360}$ test images. White arrows point to the predictions of vehicle headings.

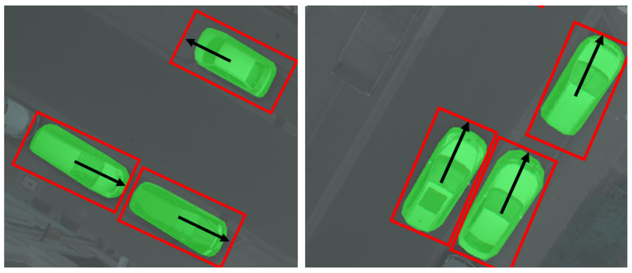
Figure 4. Qualitative results of V_{360} on UAV$_{360}$ test images. Vehicle masks and oriented bounding boxes with heading are shown.

model	Precision	Recall	F_1	mAP
V_{360}	84.6	72.9	78.3	47.1
V_b	81.9	73.7	77.6	46.1

Table 3. Precision, Recall, F_1 score and mAP for object detection on the UAV$_{360}$ dataset, achieved by the variant V_{360} of our method and the baseline V_b.

In this set of experiments, we also evaluate the estimated rotation angles of the bounding boxes. Figure 5 shows the cumulative histogram of the absolute differences between the predicted

model	Precision	Recall	F_1	mAP
V_{360}	87.4	77.6	82.2	51.3
V_b	84.2	75.5	79.6	49.9

Table 4. Precision, Recall, F_1 score and mAP for the binary masks predicted on the the UAV$_{360}$ dataset by the variant V_{360} of our method and the baseline V_b.

and the reference angles. The results show that the majority (around 80%) of the predicted angles are close to the true ones. However, the histogram also shows that the majority of heading angle errors are close to $\pm 180^o$, which corresponds to cases in which the rotation refers to the negative driving direction. On the one hand, this may be due to the difficulty of the problem: for some types of cars it may be difficult to infer the driving direction because they appear almost symmetrical in a near-nadir view. On the other hand, this may also be due to the fact that in the training process, the large majority of samples had angles only defined in the interval $(0, \pi]$; further tests using an enlarged dataset with reference angles in $(0, 2\pi]$ will show whether this assumption is correct. Figure 3 shows qualitative results of V_{360} on UAV$_{360}$ test images, depicting predicted bounding box and heading for each detected vehicle instance, and the white arrows in the figure point to the wrong predictions of the headings.

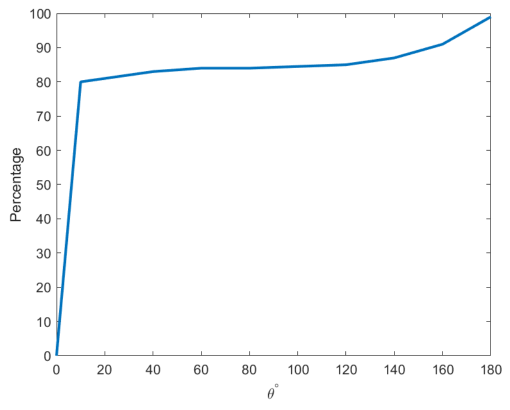

Figure 5. Cumulative histogram of absolute differences between estimated and reference angles. The abscissa gives the absolute value of the angle difference in [degrees].

5. CONCLUSION

In this paper, we have proposed a CNN-based method for vehicle instance segmentation using rotated bounding boxes to represent vehicles. The method achieved reasonably good results in object detection and in the prediction of binary masks when applied to data acquired from different platforms, including UAV. An evaluation of the predicted orientation showed that the correct orientation could be predicted in a large variety of the cases, but some errors still remain.

In future work, we plan to increase the dataset with angles in the range $(0, 2\pi]$ and see whether this will improve the results of the prediction. We slso want to analyse how the parameters of the RRPN can affect the results. Our ultimate goal is to build a method for tracking vehicles over time in the UAV images, for which the predicted rotations will be particularly useful. The tracking method does not only consider the observations from the UAV, but also street views acquired by collaborating (potentially autonomous) cars (Coenen and Rottensteiner, 2019). In this context, we also plan to differentiate between different vehicle types, which would lead to prior information about the shape of a vehicle for 3D reconstruction from the UAV images.

6. ACKNOWLEDGEMENTS

This work was supported by the German Research Foundation (DFG) as a part of the Research Training Group i.c.sens [GRK2159].

REFERENCES

Bolya, D., Zhou, C., Xiao, F., Lee, Y. J., 2019. YOLACT: Real-time instance segmentation. *IEEE International Conference on Computer Vision (ICCV)*, 9157–9166.

Coenen, M., Rottensteiner, F., 2019. Probabilistic vehicle reconstruction using a multi-task cnn. *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 822–831.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1, 886–893.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 248–255.

Ding, J., Xue, N., Long, Y., Xia, G.-S., Lu, Q., 2019. Learning RoI transformer for detecting oriented objects in aerial images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2849–2858.

Girshick, R., 2015. Fast R-CNN. *IEEE International Conference on Computer Vision (ICCV)*, 1440–1448.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 580–587.

Han, J., Ding, J., Xue, N., Xia, G.-S., 2021. Redet: A rotation-equivariant detector for aerial object detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2786–2795.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)*, 2980–2988.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Huang, Z., Huang, L., Gong, Y., Huang, C., Wang, X., 2019. Mask scoring R-CNN. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6409–6418.

Jiang, Y., Zhu, X., Wang, X., Yang, S., Li, W., Wang, H., Fu, P., Luo, Z., 2017. R2CNN: Rotational region CNN for orientation robust scene text detection. *ArXiv*, abs/1706.09579.

Kuo, W., Angelova, A., Malik, J., Lin, T.-Y., 2019. ShapeMask: Learning to segment novel objects by refining shape priors. *IEEE International Conference on Computer Vision (ICCV)*, 9207–9216.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2117–2125.

Liu, K., Mattyus, G., 2015. Fast multiclass vehicle detection on aerial images. *IEEE Geoscience and Remote Sensing Letters*, 12(9), 1938-1942.

Liu, Z., Hu, J., Weng, L., Yang, Y., 2017. Rotated region based cnn for ship detection. *IEEE International Conference on Image Processing (ICIP)*, 900–904.

Liu, Z., Wang, H., Weng, L., Yang, Y., 2016. Ship rotated bounding box space for ship extraction from high-resolution optical satellite images with complex backgrounds. *IEEE Geoscience and Remote Sensing Letters*, 13(8), 1074-1078.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3431–3440.

Lowe, D., 1999. Object recognition from local scale-invariant features. *IEEE International Conference on Computer Vision (ICCV)*, 2, 1150–1157.

Moranduzzo, T., Melgani, F., 2014. Automatic car counting method for unmanned aerial vehicle images. *IEEE Transactions on Geoscience and Remote Sensing*, 52(3), 1635-1647.

Mou, L., Zhu, X. X., 2018. Vehicle instance segmentation from aerial image and video using a multitask learning residual fully convolutional network. *IEEE Transactions on Geoscience and Remote Sensing*, 56(11), 6699-6711.

Ojala, T., Pietikainen, M., Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 971-987.

Pinheiro, P., Collobert, R., Dollar, P., 2015. Learning to segment object candidates. *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS)*, 2, 1990–1998.

Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137-1149.

Schön, S., Brenner, C., Alkhatib, H., Coenen, M., Dbouk, H., Garcia-Fernandez, N., Fischer, C., Heipke, C., Lohmann, K., Neumann, I., Nguyen, U., Paffenholz, J.-A., Peters, T., Rottensteiner, F., Schachtschneider, J., Sester, M., Sun, L., Vogel, S., Voges, R., Wagner, B., 2018. Integrity and collaboration in dynamic sensor networks. *Sensors*, 18(7). Paper 2400.

Shao, W., Yang, W., Liu, G., Liu, J., 2012. Car detection from high-resolution aerial imagery using multiple features. *2012 IEEE International Geoscience and Remote Sensing Symposium*, 4379–4382.

Wegner, J.-D., Rottensteiner, F., Sohn, G., Gerke, M., 2017. The ISPRS 2d semantic labelling challenge. www2.isprs.org/commissions/comm2/wg4/benchmark/semantic-labeling (accessed 17/01/2022).

Xia, G.-S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., Zhang, L., 2018. DOTA: A large-scale dataset for object detection in aerial images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3974–3983.

Yang, X., Liu, Q., Yan, J., Li, A., Zhang, Z., Yu, G., 2019. R3det: Refined single-stage detector with feature refinement for rotating object. *arXiv preprint arXiv:1908.05612*, 2(4), 2.

Yi, J., Wu, P., Liu, B., Huang, Q., Qu, H., Metaxas, D., 2021. Oriented object detection in aerial images with box boundary-aware vectors. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2150–2159.

Zhao, Z.-Q., Zheng, P., Xu, S.-t., Wu, X., 2019. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11), 3212–3232.