A Lightweight Keypoint-Based Oriented Object Detection of Remote Sensing Images

Yangyang Li, Heting Mao, Ruijiao Liu *, Xuan Pei, Licheng Jiao and Ronghua Shang

Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, International Research Center for Intelligent Perception and Computation, Joint International Research Laboratory of Intelligent Perception and Computation, School of Artificial Intelligence, Xi’an Jiaotong University, Xi’an 710071, China; yyli@xidian.edu.cn (Y.L.); hetingmao@stu.xidian.edu.cn (H.M.); xuanpei@stu.xidian.edu.cn (X.P.); lchjiao@mail.xidian.edu.cn (L.J.); rhshang@mail.xidian.edu.cn (R.S.)

* Correspondence: rj_liu@stu.xidian.edu.cn

Abstract: Object detection in remote sensing images has been widely used in military and civilian fields and is a challenging task due to the complex background, large-scale variation, and dense arrangement in arbitrary orientations of objects. In addition, existing object detection methods rely on the increasingly deeper network, which increases a lot of computational overhead and parameters, and is unfavorable to deployment on the edge devices. In this paper, we proposed a lightweight keypoint-based oriented object detector for remote sensing images. First, we propose a semantic transfer block (STB) when merging shallow and deep features, which reduces noise and restores the semantic information. Then, the proposed adaptive Gaussian kernel (AGK) is adapted to objects of different scales, and further improves detection performance. Finally, we propose the distillation loss associated with object detection to obtain a lightweight student network. Experiments on the HRSC2016 and UCAS-AOD datasets show that the proposed method adapts to different scale objects, obtains accurate bounding boxes, and reduces the influence of complex backgrounds. The comparison with mainstream methods proves that our method has comparable performance under lightweight.

Keywords: remote sensing image; arbitrary-oriented object detection; lightweight network; knowledge distillation

1. Introduction

With the rapid development of remote sensing technology, the quantity and quality of remote sensing images have been significantly improved, and the remote sensing interpretation has attracted more and more attention. Image classification, segmentation, detection, tracking, etc. in remote sensing images have gradually become hot topics. The object detection of remote sensing images plays a vital role in military, intelligent supervision, urban planning, etc. In this paper, we mainly studied object detection for optical remote sensing images based on deep learning.

Artificial intelligence is a data-driven technology that has been widely used in computer vision. Compared with the traditional models in the past, deep learning-based methods have great advantages, such as strong adaptability, high accuracy, and non-artificial design features. Deep learning algorithms can autonomously extract features such as object texture from data distribution and generate high-level semantic information. Benefiting from the development of the network and a large number of publicly available natural image datasets, such as MSCOCO [1] and PASCAL VOC [2], many object detection methods based on deep learning have achieved great success in the natural scene [3–10].

However, object detection algorithms that have achieved good performance on natural images cannot be directly applied to optical remote sensing images. This is because the difference between remote sensing images and natural images are very significant.
The most obvious difference between them is the shooting angle. Specifically, the remote sensing image is taken from a bird’s-eye view, which usually captures the top information of the object, while the natural image is taken from the front, which usually captures the outline of the object. Because of the different angles, remote sensing images usually have some characteristics that are different from natural images:

1. Scales of objects vary greatly—since a remote sensing image is taken from a long distance and wide angle, an image will contain objects with large-scale differences;
2. Objects with arbitrary orientations—due to the special imaging perspective, the objects in the remote sensing image have any direction;
3. Complex background—many interesting objects in remote sensing images are usually surrounded by complex backgrounds, which can seriously interfere with the detection of objects.

These characteristics make object detection in remote sensing images a particularly challenging task. For objects with arbitrary orientations, the horizontal bounding box will cause misalignment between the detected bounding box and the object [11]. To solve this problem, oriented bounding boxes are preferred for capturing objects in remote sensing images. Current oriented object detection methods are mainly originated from anchor-based detectors. Generally, those detectors generate oriented anchor boxes and then use the anchor boxes as a reference to learn the direction information and regress the offsets between the object box and the anchor box parameters (such as R2PN [12], R-DFPN [13]), or after generating the region-of-interest (ROI), regress the angle parameters to achieve oriented object detection (such as ROI transformer [11]). Although these methods accomplish the detection of oriented objects in remote sensing images, they are still anchor-based and share the same drawbacks with anchor-based detectors. For example, the design of the anchor box is complicated. The aspect ratios and the size of the anchor boxes need to be carefully adjusted. In addition, the sharp imbalance between the positive and negative anchor boxes can lead to slow training and suboptimal performance.

In recent years, researchers have developed keypoint-based object detectors [14–16] to overcome the shortcomings of anchoring-based solutions in horizontal object detection tasks. The keypoint-based object detector [17,18] captures the object by detecting the keypoint, thereby providing an anchor-free way. In particular, these methods detect the corner points or center points of the bounding boxes and then extract the size of the bounding boxes from these points. Cornernet [17] is one of the pioneers who proposed keypoint-based detection methods in horizontal object detection tasks. It uses heatmaps to capture the top left and bottom right corner points of the horizontal bounding box and then groups the corners. CenterNet [18] proposed by Duan detects corner points and center points. ExtremeNet [19] locates the extreme points and center points of the bounding boxes. These two methods finally group the points of the box by the center information. Zhou’s CenterNet [20] is a faster prediction method because there is no post-processing, and only recovers the width and height of the bounding box at the center point. Keypoint-based detectors have achieved great success in horizontal detection tasks, and have advantages over anchor-based detectors in speed and accuracy. However, keypoint-based detectors are rarely used in oriented object detection.

In many applications, the lightweight nature of models is a key requirement, and fundamentally it has a competitive relationship with accuracy. Although object detection in remote sensing images has made great progress, they rely on the increasingly deeper network, which increases a lot of computational expenses and parameters. To solve the problem of over-parameterization, some previous works have designed new network structures, such as fully convolutional networks, or lightweight networks [21]. These networks have made progress to a certain extent, but they need to be carefully designed and adjusted. Some image classification uses model compression, which decomposes the weights of each layer, and then reconstructs or fine-tunes layer by layer to restore partial accuracy [22–25]. However, there is often a gap between the accuracy of the original model and the compressed model, especially those which will become larger when using...
model compression to solve more complex problems, such as remote sensing image object detection. On the other hand, the research of knowledge distillation shows that imitating the behavior of deeper or more complex network models to obtain low-level models or have a compressed effect can make up for part or all of the accuracy gap [26–28]. However, these results only show up on classification issues. Object detection is a more complex task, which includes two subtasks: classification and bounding box regression.

Inspired by the above research, we propose a keypoint-based oriented object detector for remote sensing images and lighten this detector through the knowledge distillation method. We use BBAAVectors as a benchmark and make improvements on it. First, a semantic transfer block is proposed, which uses semantic information as a guide to refine features. It solves the problem of excessive noise information in features due to complex backgrounds. In addition, considering the variety of object sizes in the remote sensing image, we propose the adaptive Gaussian kernel to generate adaptive Gaussian heatmaps to locate and classify objects. For the problem that the neural network is always over-parameterized, we propose the distillation loss associated with object detection in remote sensing images to obtain a lightweight student network. Finally, the effectiveness of this method is verified on two datasets. The contributions of this paper are summarized as follows:

1. In order to avoid the noise caused by the complex background of remote sensing images, we proposed a semantic transfer block to refine the low-level features when fusing high-level features and low-level features. The refined low-level features not only reduce noise but also restore the semantic information of the fused features;
2. Considering that the scale of objects in remote sensing images varies greatly, the adaptive Gaussian kernel is proposed to produce adaptive Gaussian heatmaps to locate and classify objects. Compared with the traditional heatmap, our improved heatmap adapts to objects of different sizes and avoids the fuzzy samples caused by traditional heatmap labels;
3. To solve the problem that the object detection model of remote sensing images is always over-parameterized, we propose a distillation framework for the keypoint-based object detection network. For the four heads of the network, we calculate the distillation loss based on the soft labels and combine it with the loss using hard labels to distillation the student network.

2. Proposed Methods

In this section, we will describe the various parts of our pipeline in detail. Figure 1 shows the overall framework of our method. The network is built on a U-shaped structure, and the feature map is upsampled at the top of the backbone. In the process of upsampling, we use the semantic transfer block (STB) to introduce the deep semantic information into the low-level features as a guide to generate the low-level features with semantic guidance. Then, we merge the deep features with the low-level features through jump connections, sharing high-level semantic information and low-level and more refined details. Using $3 \times 3$ convolutions and $1 \times 1$ convolutions, the fused feature maps $F \in \mathbb{R}^{C \times \frac{H}{s} \times \frac{W}{s}}$ are converted to four prediction branches: Heatmap ($P \in \mathbb{R}^{K \times \frac{H}{s} \times \frac{W}{s}}$), Offset ($E \in \mathbb{R}^{2 \times \frac{H}{s} \times \frac{W}{s}}$), Box Parameter ($B \in \mathbb{R}^{10 \times \frac{H}{s} \times \frac{W}{s}}$), and Orientation ($\theta \in \mathbb{R}^{1 \times \frac{H}{s} \times \frac{W}{s}}$), where $K$ is the number of classes and $s = 4$ is the scale. Finally, we use knowledge distillation to obtain a lightweight keypoint-based oriented object detector for remote sensing images. Specifically, we regard the network which backbone is ResNet50 as the teacher network and the network which backbone is ResNet18 as the student network. After inputting images into the two networks, we use the final outputs of the teacher network as soft labels to assist student network learning. In this process, the mean square error (MSE) loss function is used to align the heatmap and orientation branch, and the $L_1$ loss is used to align the offset and bounding box parameters. In addition, we also use the ground-truths of the dataset as the hard labels and use them to train the student detection network.
Figure 1. The overall framework of our method: (a) a keypoint-based oriented object detection network for remote sensing images. STB denotes the semantic transfer block; (b) the heatmap and orientation from teacher and student networks are aligned via the mean square error, and the heatmap and offset and box parameters are aligned via the $L_1$ loss.

2.1. Keypoint-Based Object Detection

2.1.1. Semantic Transfer Block

The backbone network uses a U-shaped structure, similar to U-Net [29]. Although this U-shaped structure has achieved great success, high-level features and low-level features are simply merged through simple jump connections or horizontal connections, but its working mechanism is still unknown. Whether this simple fusion is really helpful for remote sensing image object detection is worthy of further study. The high-level and low-level features are complementary in nature. The low-level features have rich spatial details but lack semantic information and vice versa. Consider an extreme case where pure low-level features only contain information such as points, lines or edges. Intuitively, fusing high-level features with this purely low-level feature is hardly helpful, because low-level features contain too much noise, especially in scenes with complex backgrounds such as remote sensing images, and cannot provide enough high-resolution semantic information guides feature fusion. On the contrary, if the low-level features contain relatively clear semantic boundaries, it will be more helpful to merge the low-level features with more semantic information with the high-level features. Similarly, high-level semantic features with little spatial information cannot make full use of low-level texture features. However, by introducing additional high-resolution features, high-level semantic features may have the opportunity to refine their information by aligning the boundary information in related low-level features. Therefore, we propose the semantic transfer block (STB) to provide a more effective feature fusion method.
\[ y_l = \text{Upsample}(y_{l+1}) + \mathcal{F}(x_l, x_{l+1}) \]  

(1)

where \( l \) represents the level of the feature map. The meaning of Formula (1) is to introduce more semantic information from high-level features to guide feature fusion. The function \( \mathcal{F}(\cdot) \) represents the semantic migration module, and its design details are shown in Figure 2. Specifically, we first upsample the deep features to the same size as the low-level features through bilinear interpolation, refine the upsampled feature maps through a \( 3 \times 3 \) convolution, and then combine the feature maps with the low-level features. Finally, a \( 1 \times 1 \) convolutional layer is used to refine the channel of fusion features. In this process, batch normalization (BN) and ReLU activation functions are used in the hidden layer. We apply this component for level 2–4 features (see Figure 1).

![Figure 2. The building block of the semantic transfer block. The \( \odot \) denotes element-wise addition.](image)

To better prove the above point, we visualize the feature map, as shown in Figure 3. Figure 3a shows a low-level feature map in the backbone. Figure 3c shows a low-level feature map after STB processing. Figure 3b shows a merged feature map that combines the original low-level feature with a high-level feature through jump connections. Figure 3d shows a STB processed low-level feature with a high-level feature through jump connection. It can be seen from the figure that the low-level features of the backbone have too much noise information, which leads to the feature maps obtained by jump connection being too fuzzy. This shows that if the low-level features contain less semantic information, the jump connection cannot restore the semantics. In contrast, the low-level features processed by STB introduce more semantic information and reduce more noise, resulting in the merged feature maps restore the semantics well.
Figure 3. An example of feature maps: (a) a low-level feature map in the backbone; (b) a merged feature map that combines an original low-level feature with a high-level feature through jump connection; (c) a low-level feature map after STB processing; (d) a merged feature map that merges a STB processed low-level feature with a high-level feature through jump connection.

2.1.2. Adaptive Gaussian Kernel

Heatmaps are usually used to locate specific keypoints of the input image, such as human joints and facial feature points. In this paper, the heatmap is used to detect the center point of the oriented object in the remote sensing image. Specifically, the heatmap \( P \in \mathbb{R}^{K \times H \times W} \) applied in this paper has \( K \) channels, and each channel corresponds to an object category. The mapping on each channel is passed through a sigmoid function. We used the heatmap value predicted by the specific center point as the confidence level for object detection.

Assuming that \( c = (c_x, c_y) \) represents the center point of the directed bounding box, a two-dimensional Gaussian kernel needs to be set around each center point to form the heatmap. The original 2D Gaussian is defined as follows:

\[
G(x, y) = \exp\left(-\frac{(x - c_x)^2 + (y - c_y)^2}{2\sigma^2}\right)
\]

where \((x, y)\) is a pixel on the heatmap, and \(\sigma\) is a standard deviation adapted to the size of the bounding box. Since the standard deviation corresponding to the width and height of the bounding box in the original Gaussian map is equal, the situation shown in Figure 4a may occur, which will cause the division of positive and negative samples to be blurred, and will affect the training of the model.

In other words, we use a 2D Gaussian to generate a heatmap to locate the center point of the object. The center point of the object has the highest confidence, and the confidence of the rest of the bounding box will gradually decrease according to the Gaussian distribution. The regions outside the bounding box are all negative samples. In theory, the confidence of the corresponding position of the heatmap should be zero. However, in the case of Figure 4a, the confidence of the mapping of some areas beyond the bounding box is not zero.
These samples are fuzzy samples for the model, which will cause the model to calculate the loss during training inaccurate weighting, thereby affecting the final performance.

![Figure 4](image_url)

**Figure 4.** The heatmaps of different Gaussian kernel: (a) the Gaussian kernel; (b) the adaptive Gaussian kernel.

Therefore, we propose an adaptive Gaussian kernel (AGK), as shown in Figure 4b. The specific definition is as follows:

$$ AG(x, y) = \exp \left(-\frac{(x - c_x)^2}{2\sigma_w^2} - \frac{(y - c_y)^2}{2\sigma_h^2}\right) $$  \hspace{1cm} (3)

where $\sigma_w = \frac{\gamma_w \cdot b}{6}$ and $\sigma_h = \frac{\gamma_h \cdot b}{6}$ are, respectively, proportional to the width and height of the bounding box. $\gamma$ is a hyperparameter and we set $\gamma = 0.54$ in this paper based on experience. Too large a value of $\gamma$ would the Gaussian map exceeding the bounding box. The adaptive Gaussian heatmap generated by the adaptive Gaussian kernel is shown in Figure 5.

![Figure 5](image_url)

**Figure 5.** The adaptive Gaussian heatmap generated by the adaptive Gaussian kernel.

When training the heatmap, the peak point of the Gaussian distribution (that is, the center point $c$ of the object) is regarded as a positive sample, and any other pixels are regarded as a negative sample. Due to the imbalance of positive and negative samples, it will be difficult to directly learn the central point. Therefore, in this article, we use the variant focal loss [17] to train the heatmap, which is defined as follows:

$$ L_h = -\frac{1}{N} \sum_i \left\{ \begin{array}{ll} (1 - p_i)^a \log(p_i) & \text{if } \hat{p}_i = 1 \\ (1 - \hat{p}_i)^{\beta} p_i^\alpha \log(1 - p_i) & \text{otherwise} \end{array} \right. $$  \hspace{1cm} (4)

where $\hat{p}$ and $p$ represent the object heatmap value and the predicted heatmap, respectively, and $i$ represents the pixel position on the feature map, $N$ represents the number of ob-
jects, and $\alpha$ and $\beta$ are the hyperparameters used to control the contribution of each pixel. According to the experience in [17], we choose $\alpha = 2$ and $\beta = 4$.

2.1.3. Offset

In the inference process of the model, we need to extract the peak point of the predicted heatmap and use it as the center location of the object. This center point is an integer, but it becomes a floating-point number after downsampling the original image. In order to compensate for this quantization error, the network needs to predict an offset $e$:

\[
e = \left( \frac{c_x}{s} - \left\lfloor \frac{c_x}{s} \right\rfloor, \frac{c_y}{s} - \left\lfloor \frac{c_y}{s} \right\rfloor \right)
\]  

(5)

The offset is optimized using the smooth $L_1$ loss [3]:

\[
L_e = \frac{1}{N} \sum_{n=1}^{N} \text{Smooth}_{L_1}(e_n - \hat{e}_n)
\]

(6)

where $N$ denotes the number of objects, $n$ denotes the index of the object, and $\hat{e}$ and $e$ denote the object offset and the predicted offset, respectively. The expression of smooth $L_1$ is as follows:

\[
\text{Smooth}_{L_1}(x) = \begin{cases} 
0.5x^2 & \text{if } |x| < 1 \\
|x| - 0.5 & \text{otherwise}
\end{cases}
\]

(7)

2.1.4. Orientation

There are two types of oriented bounding boxes (OBB), one is a horizontal bounding box (HBB) and the other is a rotating bounding box (RBB). When the oriented bounding box is horizontal, an accurate horizontal bounding box can be obtained through $w$ and $h$ in the bounding box parameters. When the oriented bounding box is a rotating box, the top $t$, right $r$, bottom $b$, and left $l$ vectors from the center points of the objects can describe the rotating bounding box, as shown in Figure 6. For the detector to better learn the direction information of the bounding box, a branch is introduced to predict the orientation coefficient $\theta$:

\[
\hat{\theta} = \begin{cases} 
1(\text{RBB}) & \text{IOU(HBB, OBB) < 0.95} \\
0(\text{HBB}) & \text{otherwise}
\end{cases}
\]

(8)

where the IOU is the intersection over union (IOU) between the horizontal bounding box and the oriented bounding box. In training, the orientation coefficient is optimized utilizing the standard binary cross-entropy loss function:

\[
L_\theta = -\frac{1}{N} \sum_{n=1}^{N} \left( \hat{\theta}_n \log(\hat{\theta}_n) + \left(1 - \hat{\theta}_n\right) \log(1 - \hat{\theta}_n) \right)
\]

(9)

where $\hat{\theta}$ and $\theta$ represent the object orientation coefficient and the predicted orientation coefficient.
2.1.5. Box Parameter

In order to obtain the oriented bounding box, we use four-vectors \( t, r, b, l \) to describe the oriented bounding box, as shown in Figure 6. In addition, \( w \) and \( h \) are the width and height of the external horizontal bounding box of the oriented bounding box used to convert coordinate when the predicted bounding box is horizontal. Therefore, the bounding box parameters are defined as \( B = [t, r, b, l, w, h] \), and a smooth \( L_1 \) loss is used to regress the bounding box parameter:

\[
L_b = \frac{1}{N} \sum_{n=1}^{N} \text{Smooth}_{L_1}(B_n - \hat{B}_n)
\]

where \( \hat{B} \) and \( B \) are the object bounding box parameters and the predicted bounding box parameters.

In the process of model training, the above bounding box parameter \( B \) is continuously optimized. In the inference stage, four coordinate points of the bounding box need to be obtained. First, the center point is adjusted by \( \hat{c} = c + e \), where \( e \) is the predicted offset; then, the adjusted center point is mapped back to the input image through \( \bar{c} = \hat{c} \). When the orientation coefficient \( \theta > 0.8 \), the predicted oriented bounding box is a rotating box, otherwise, the predicted oriented bounding box is horizontal. In this paper, we used the top left (tl), bottom left (bl), bottom right (br), and top right (tr) to describe the points of the bounding box after decoding. The way to decode the rotating box is as follows:

\[
\begin{align*}
\text{tl} &= (t + l) - \bar{c}, \\
\text{tr} &= (t + r) - \bar{c}, \\
\text{bl} &= (b + l) - \bar{c}, \\
\text{br} &= (b + r) - \bar{c}
\end{align*}
\]

(11)

When the oriented bounding box is horizontal, the following definition is utilized to obtain the horizontal bounding box:

\[
\begin{align*}
\text{tl} &= \left( \bar{c}_x - \frac{w}{2}, \bar{c}_y - \frac{h}{2} \right), \\
\text{tr} &= \left( \bar{c}_x + \frac{w}{2}, \bar{c}_y - \frac{h}{2} \right), \\
\text{bl} &= \left( \bar{c}_x - \frac{w}{2}, \bar{c}_y + \frac{h}{2} \right), \\
\text{br} &= \left( \bar{c}_x + \frac{w}{2}, \bar{c}_y + \frac{h}{2} \right)
\end{align*}
\]

(12)

It can be seen from the above functions that all the outputs are directly estimated from the center keypoints. The total loss is composed of the loss function of the four branches:

\[
L = L_h + L_c + L_b + L_\theta
\]

(13)
2.2. Knowledge Distillation

As can be seen from the previous section, our detector includes an encoding–decoding structure for feature extraction and four prediction branches: Heatmap, Offset, Box Parameter, and Orientation. In this section, we will use knowledge distillation to transfer knowledge from the larger teacher network into a smaller, distilled student network. Specifically, the network for which the backbone is ResNet50 as the teacher network and the network for which the backbone is ResNet18 as the student network. After inputting images into the two networks, we used the final outputs of the teacher network as soft labels to assist student network learning. In this process, the mean square error (MSE) loss function is used to align the heatmap branch and the orientation branch, and the $L_1$ loss is used to align the offset branch and bounding box parameters. In addition, we also used the ground-truth labels of the dataset as the hard labels and used them to train the student detection network.

We utilized $L_{total}$ to train the student model to align with the teacher model, which can be formulated as

$$L_{total} = L_H + L_{\Theta} + L_E + \lambda L_B$$

(14)

where $L_H$, $L_{\Theta}$, $L_E$ and $L_B$ are denote the heatmap loss, orientation loss, offset loss, and box parameter loss of the student network, which are composed of two components: the loss calculated by hard labels and the knowledge distillation loss calculated by soft labels. We introduced a hyper-parameter $\lambda$ to balance different loss, and set $\lambda = 0.1$ based on experience.

Distillation Loss

The heatmap prediction branch is responsible for the classification and localization of objects. Specifically, it predicts the center keypoint to locate the objects and takes the center keypoint as a positive sample, and the remaining points are all negative samples. Since there are multiple objects for object detection, the heatmap branch trains multiple binary classifiers. In image classification, the knowledge distillation framework is to distill the logits output of the network, which does not apply to the heatmap branch where multiple binary classifiers are trained. Therefore, in this paper, we propose the distillation loss for the alignment of the heatmap branches between the student network and the teacher network, which is defined as follows:

$$L_{h}^{soft} = \frac{1}{N} \sum_{i=1}^{N} (h_s - h_t)^2$$

(15)

Consequently, the loss function of the heatmap branch is:

$$L_H = L_h^{hard}(h_s, y_h) + L_h^{soft}(h_s, h_t)$$

(16)

where $y_h$ represents the label of the heatmap branch. $h_s$ and $h_t$ are the predicted value of the heatmap yield by student network and teacher network. $L_h^{hard}$ and $L_h^{soft}$ are the losses calculated by the student network using hard labels and soft labels.

As is well known, a deeper teacher network can better adapt to the training dataset and perform well on the test dataset. Since the soft labels contain the category information learned by the teacher network, the student network can inherit this information by learning soft labels.

Since the prediction of the orientation is also a two-class classification, we use the same distillation loss as the heatmap to learn the knowledge of the orientation of the teacher network. The specific definition is as follows:
The distillation loss of these two branches adopts MSE loss because these two branches pay more attention to abnormal cases. The heatmap branch only takes the center point as a positive sample and treats other points as anomalies, so it needs to be very accurate to determine which points are positive samples or not. Similarly, for the orientation branch, if the output is greater than a certain threshold, the oriented bounding box is considered to be rotated; otherwise, the output is equivalent to an abnormal value when it is less than the threshold. In summary, the network needs to adapt well to these two situations, so it is necessary to pay more attention to some outliers.

The orientation of the student network is optimized by the following:

$$L_{\theta} = L_{\text{hard}}^{\theta}(\theta_s, y_\theta) + L_{\theta}^{\text{soft}}(\theta_s, \theta_t)$$ (18)

where $y_\theta$ represents the orientation coefficient label, $\theta_s$ and $\theta_t$, respectively, represent the predicted output of the student network and the teacher network in orientation branch, $L_{\theta}^{\text{hard}}$ is the loss calculated by the student network using the hard labels, and $L_{\theta}^{\text{soft}}$ is the distillation loss obtained through soft labels.

For the offset and box parameter, we use the $L_1$ loss as the distillation loss:

$$L_{e}^{\text{soft}} = \frac{1}{N} \sum_{i=1}^{N} |e_s - e_t|$$ (19)

$$L_{b}^{\text{soft}} = \frac{1}{N} \sum_{i=1}^{N} |b_s - b_t|$$ (20)

When the center keypoint and orientation are predicted, the offset and the box parameter do not need to pay much attention to the outliers, so we use $L_1$ to calculate their distillation loss. Specifically, the $L_1$ loss will result in sparse features and reset most useless weights to zero. Therefore, these two branches only focus on the difference between the prediction of the student network and the output of the teacher network.

The offset and box parameter of the student network is trained by the following definitions:

$$L_E = L_{e}^{\text{hard}}(e_s, y_e) + L_{e}^{\text{soft}}(e_s, e_t)$$ (21)

$$L_B = L_{b}^{\text{hard}}(b_s, y_b) + L_{b}^{\text{soft}}(b_s, b_t)$$ (22)

where $y_e$ and $y_b$ denote the offset label and the bounding box parameter label, and $e_s$, $b_s$, $e_t$ and $b_t$ denote the output of the student network and the teacher network in these two branches. $L_{e}^{\text{hard}}$ and $L_{b}^{\text{hard}}$ are the loss calculated by hard labels. $L_{e}^{\text{soft}}$ and $L_{b}^{\text{soft}}$ are the distillation losses calculated based on soft labels.

3. Experiments and Results

In this section, we will introduce the datasets, implementation details and results in the experiment. All experiments are performed on a piece of NVIDIA Geforce RTX 2080Ti 11GB with Pytorch framework.

3.1. Datasets

We evaluated our approach on two public remote sensing datasets: HRSC2016 [30] and UCAS-AOD [31], in which ground-truth labels are all marked with oriented bounding boxes. The datasets used for the experiments in this paper are briefly introduced as follows:
HRSC2016 is a high-resolution optical remote sensing dataset for ship recognition. All the images are collected from Google Earth and their resolutions are between 2 m and 0.4 m. The image sizes range from 300 × 300 to 1500 × 900 and most of them are larger than 1000 × 600. Each sample in HRSC2016 is annotated with the bounding box, rotated bounding box, and ship head location, with three-level classes including ship, ship category, and ship types. The dataset contains 617 training images and 438 test images. We trained the network on the training set and evaluated the performance on the test set.

UCAS-AOD is a vehicle dataset collected from Google Earth, for which the objects are airplanes and vehicles under aerial images. The dataset contains a total of 1000 aircraft images and 510 vehicle images, and its image data are divided into three parts: CAR, PLANE, and NEG. The positive image is named with P+number, the negative image is named with N+number, and all image formats are in PNG format. The ground-truth label includes the coordinates, height, width, and rotation angle of the object, and its format is in TXT format. Because the dataset was not formally divided into the training set and test set, we randomly divided it into 50%, 20%, and 30% as the training set, validation set, and test set, respectively. In our experiment, all images are adjusted to 608 × 608.

3.2. Implementation Details

3.2.1. Training and Inference

The backbone of the teacher network uses the first five convolutions of ResNet50 pre-trained on the ImageNet dataset. The remaining weights are initialized under Pytorch’s default settings. We reshape the images to 608 × 608 in the training and inference stage, and the final output size is 152 × 152. In addition, we used standard data augmentations, including random flipping and random cropping in the scale range [0.9, 1.1]. In the training process, the Adam algorithm [32] was adopted with an initial learning rate of $1.25 \times 10^{-4}$ to optimize our teacher network and an exponential decay strategy was used to update the learning rate. The batch size was set to 10, and 100 epochs were trained on both datasets.

For the student network, it first used the first five convolutions of ResNet18 as the backbone, and the initialization of the remaining network weights, data processing, and optimization strategies are the same as the teacher network. The difference is that the distilled student network is trained with a batch size of 16 and 100 epochs on both datasets.

3.2.2. Evaluation Indicators

In order to quantitatively evaluate our model’s detection accuracy, we used average accuracy (AP) and mean average accuracy, which are two widely used evaluation indicators [33].

AP is the average precision of the object in the range of recall = [0, 1], and is generally the area under precision-recall curve (PRC). PRC can be obtained through four evaluation components: true positive (TP), false positive (FP), false negative (FN), and true negative(TN). Recall (R), precision (P), and AP are defined as

\[
\text{precision} = \frac{TP}{TP + FP} \\
\text{recall} = \frac{TP}{TP + FN} \\
AP = \int_{0}^{1} P(R)dR
\]

where TP, FP, and FN represent the number of correctly detected objects, the number of incorrectly detected objects, and the number of non-detected objects, respectively. mAP is adopted for multi-class evaluation, which is the average value of AP values for all classes.

For the quantitative evaluation of the model efficiency, we measured the model size and the calculation complexity [34]. The model size was expressed by the number of network parameters. Furthermore, the calculation complexity was represented by the sum
of the floating-point operations (FLOPs) in once forward on a fixed input size. The number of network parameters and FLOPs were calculated by the Pytorch-OpCounter package.

3.3. Peer Methods Comparison

3.3.1. Results on HRSC2016

In order to show the performance of our proposed method, we compared our results with R$^2$CNN [35], RRPN [36], ROI Transformer [11], and BBAVectors [37]. The performance of these methods is shown in Table 1. Compared with the existing published oriented object detection models, our network gains 14.65%, 8.67%, 1.52%, and 1.4% improvements over R$^2$CNN, RRPN, ROI Transformer, and BBAVectors on the HRSC2016 dataset, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Image Size</th>
<th>AP</th>
<th>Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>R$^2$CNN</td>
<td>ResNet-101</td>
<td>800 × 800</td>
<td>73.07</td>
<td>158 MB</td>
</tr>
<tr>
<td>RRPN</td>
<td>ResNet-101</td>
<td>800 × 800</td>
<td>79.05</td>
<td>181 MB</td>
</tr>
<tr>
<td>ROI Transformer</td>
<td>ResNet-101</td>
<td>512 × 800</td>
<td>86.20</td>
<td>74.12 MB</td>
</tr>
<tr>
<td>BBAVectors</td>
<td>ResNet-50</td>
<td>608 × 608</td>
<td>86.32</td>
<td>53.43 MB</td>
</tr>
<tr>
<td>Teacher</td>
<td>ResNet-50</td>
<td>608 × 608</td>
<td>87.72</td>
<td>90.60 MB</td>
</tr>
<tr>
<td>Undistilled Student</td>
<td>ResNet-18</td>
<td>608 × 608</td>
<td>84.12</td>
<td>13.52 MB</td>
</tr>
<tr>
<td>Distilled Student</td>
<td>ResNet-18</td>
<td>608 × 608</td>
<td>87.34</td>
<td>15.84 MB</td>
</tr>
</tbody>
</table>

Table 1. Overall performance evaluation of different methods on HRSC2016 datasets. Bold numbers in the table mean the best results.

Note that the backbone used by our teacher network is ResNet50, while the backbone used by R$^2$CNN, RRPN, and ROI Transformer is ResNet101, however, the results of our method are better than these networks. In addition, most of the image sizes used by the compared methods are larger than our network, and the larger image size is helpful for the detection of remote sensing image objects. However, it can be seen from Table 1 that our method can still outperform other methods at an image size of 608 × 608. BBAVectors is the benchmark of our method. Under the same settings, the results of our method are better than the benchmark network, which shows the effectiveness of our method.

Table 1 also shows the comparison between the distilled student network and other methods on the HRSC2016 dataset. The detection results of the undistilled student network are only better than R$^2$CNN and RRPN. However, the distilled student network gains 8.29%, 1.14%, 1.02%, and 3.22% improvements in AP compared with RRPN, ROI Transformer, BBAVectors, and undistilled student networks. The accuracy of the distilled student network is very close to that of the teacher network, but its parameter is 15.84 MB, which is only 1/6 of the teacher network. Note that the distilled student network can still achieve better performance than other methods when the size of the input image is relatively smaller and the backbone network is shallower than other methods. This shows the effectiveness of our method.

3.3.2. Results on UCAS-AOD

To further demonstrate the performance of our method, we compared it with the directed object detection method on the UCAS-AOD dataset. The results are shown in Table 2. Compared with R-YOLOv3 [38], R-RetinaNet [39], Faster RCNN [10], and BBAVectors [37], our teacher network achieves 8.29%, 1.14%, 1.02%, and 3.22% advances in mAP. By analyzing the AP value, it can be obtained that the teacher network has achieved a significant improvement on the vehicle, and achieved a second-best performance on the aircraft, which shows that it is relatively robust to a small object such as a vehicle. The distilled student network is very close to the teacher network in mAP, and at the same time, gains 7.85%, 2.36%, 1.57%, 1.37%, and 1.74% compared with R-YOLOv3, R-RetinaNet, Faster RCNN, BBAVectors, and the undistilled student network, respectively. The distilled
student network is better than the other methods and is comparable to the teacher network in AP.

### Table 2. Overall performance evaluation of different methods on UCAS-AOD datasets. Bold numbers in the table mean the best results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Image Size</th>
<th>Plane</th>
<th>Vehicle</th>
<th>mAP</th>
<th>Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-YOLOv3</td>
<td>DarkNet53</td>
<td>800 x 800</td>
<td>89.52</td>
<td>74.63</td>
<td>82.08</td>
<td>62.40 MB</td>
</tr>
<tr>
<td>R-RetinaNet</td>
<td>ResNet-50</td>
<td>800 x 800</td>
<td><strong>90.51</strong></td>
<td>84.64</td>
<td>87.57</td>
<td>36.42 MB</td>
</tr>
<tr>
<td>Faster RCNN</td>
<td>ResNet-50</td>
<td>800 x 800</td>
<td>89.86</td>
<td>86.87</td>
<td>88.36</td>
<td>60.21 MB</td>
</tr>
<tr>
<td>BBAVectors</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td>90.26</td>
<td>86.87</td>
<td>88.56</td>
<td>53.43 MB</td>
</tr>
<tr>
<td>Teacher</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td><strong>90.43</strong></td>
<td><strong>89.46</strong></td>
<td><strong>89.95</strong></td>
<td>90.60 MB</td>
</tr>
<tr>
<td>Undistilled Student</td>
<td>ResNet-18</td>
<td>608 x 608</td>
<td>90.27</td>
<td>86.11</td>
<td>88.19</td>
<td><strong>13.52 MB</strong></td>
</tr>
<tr>
<td>Distilled Student</td>
<td>ResNet-18</td>
<td>608 x 608</td>
<td><strong>90.51</strong></td>
<td>89.36</td>
<td>89.93</td>
<td>15.84 MB</td>
</tr>
</tbody>
</table>

#### 3.4. Ablation Study

In order to verify the effectiveness of the semantic transfer block and adaptive Gaussian kernel, we conducted ablation experiments on the test set of the UCAS-AOD dataset. In the experiment, we used AP and mAP as evaluation indicators. We chose BBAVectors as the baseline, and its backbone is ResNet50. For the sake of fairness, all experimental data and parameter settings are strictly consistent. Table 3 summarizes the results of models with different modules on our UCAS-AOD dataset. It can be seen from the table that the baseline in which the semantic transfer block is added increases mAP by 1.11%, replacing the Gaussian heatmap in the baseline with an adaptive Gaussian heatmap generated by the adaptive Gaussian kernel, increases mAP by 1.15%, and after adding these two modules, the network achieved a 1.39% improvement on mAP. The results show that both the STB and AKG can improve the detection results of the model, especially the small object of vehicles, which proves the validity of our method.

### Table 3. Results of ablation studies on different components. +STB means that we add the proposed semantic transfer block to the baseline network. +AGK means that we replace the Gaussian heatmap in the baseline with an adaptive Gaussian heatmap generated by the adaptive Gaussian kernel. Bold numbers in the table mean the best results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Image Size</th>
<th>Plane</th>
<th>Vehicle</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td>90.26</td>
<td>86.87</td>
<td>88.56</td>
</tr>
<tr>
<td>Baseline+STB</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td>90.33</td>
<td>89.03</td>
<td>89.67</td>
</tr>
<tr>
<td>Baseline+AGK</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td>90.41</td>
<td>89.01</td>
<td>89.71</td>
</tr>
<tr>
<td>Baseline+STB+AGK</td>
<td>ResNet-50</td>
<td>608 x 608</td>
<td><strong>90.43</strong></td>
<td><strong>89.46</strong></td>
<td><strong>89.95</strong></td>
</tr>
</tbody>
</table>

In order to further prove the benefit of knowledge distillation, we measure the calculation complexity of our model.

Table 4 shows the calculation complexity of the teacher and student networks. From Tables 1, 2 and 4, it can be seen that compared with the teacher network, the FLOPs of the student network before distillation are reduced by four times, but its accuracy is not as good as that of the teacher network. However, the distilled student network is not only as small as FLOPs as undistilled students, however, the accuracy is also closer to that of the teacher network. These data lead us to conclude that the knowledge distillation method in this paper can learn a lightweight and accurate oriented object detector in remote sensing images. The distilled student network we proposed can greatly reduce the complexity of the model with less loss of accuracy, which is conducive to practical application.
Table 4. The calculation complexity of our methods. Bold numbers in the table mean the best results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>Image Size</th>
<th>FLOPs(G)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher</td>
<td>ResNet-50</td>
<td>608 × 608</td>
<td>109.35</td>
</tr>
<tr>
<td>Undistilled Student</td>
<td>ResNet-18</td>
<td>608 × 608</td>
<td>26.36</td>
</tr>
<tr>
<td>Distilled Student</td>
<td>ResNet-18</td>
<td>608 × 608</td>
<td>26.36</td>
</tr>
</tbody>
</table>

3.5. Visualization

To visually show the performance of the teacher network, we visualized the detection results of the HRSC2016 dataset, as shown in Figure 7. It can be seen from the figure that the teacher network can detect different scales and obtain accurate bounding boxes. This shows that the adaptive Gaussian heatmap we use cannot only adapt to objects of different scales, avoid the adverse effects of fuzzy samples, but also accurately predict the center keypoints of the object. In addition, we can also see that the teacher network can precisely detect ships at the edge of the port. This is because the proposed semantic migration module provides more refined feature maps for the network. In summary, the experimental results show that the network can effectively avoid the adverse effects of background noise, adapt to objects of different proportions, and has good robustness.

The results of the teacher network on the UCAS-AOD dataset are shown in Figure 8. It can be seen that the network can accurately detect vehicles and airplanes in different orientations and scales.

![Figure 7. Detection results of the teacher network on the HRSC2016 dataset.](image)

![Figure 8. Detection results of the teacher network on the UCAS-AOD dataset.](image)

The results of the student network after distillation on the HRSC2016 and UCAS-AOD datasets are shown in Figures 9 and 10. We chose to display remote sensing images with more complex backgrounds in Figures 9 and 10. Especially in the second column of the
figures, there are several objects hidden in the environment. It can be seen from the figure that the distilled student network still retains the advantage of the teacher. Even if the object is in a complex environment, the student network can still properly detect the object. These examples indicate that the student network has learned the knowledge of the teacher network well.

Figure 9. Detection results of the student network on the HRSC2016 dataset.

Figure 10. Detection results of the student network on the UCAS-AOD dataset.

4. Discussion

By comparing and analyzing experiments, the validity of our method is verified. From the visualization of the detection results, we can see that our network can avoid the adverse effects of background noise, adapt to objects of various scales, and has good robustness.

Semantic transfer block (STB) provides a stronger feature fusion method than jump connection. This is because low-level features have the location information of the object, but lack semantic clues. In addition, the shallow feature maps in remote sensing images usually contain much noise, causing the feature maps after direct fusion to be overly blurred, as shown in Figure 3. The shallow feature maps processed by STB introduce more semantic information and reduce noise so that the fused feature map can restore the semantics well.

The proposed adaptive Gaussian kernel (AGK) can adapt to objects of different scales and eliminate the influence of fuzzy samples on model training. This is because the heatmaps generated by AGK are proportional to the width and height of the bounding box, such as that shown in Figure 4. This avoids the fact that the confidence of some locations beyond the bounding box is not zero since the original 2D Gaussian, which results in the blurring of positive and negative samples and has a negative effect on training.
The distillation loss is based on the idea of knowledge distillation in the classification task. In the classification task, the ground-truths are usually converted into one-hot encodings, which are called hard labels. Correspondingly, the outputs of the softmax function of the teacher network are called soft labels, which represent the probability distribution of the current prediction object and contain more information than hard labels. Distillation loss is to use the outputs of the four prediction branches as soft labels, so that the prediction results of the student network are aligned with the teacher, thereby obtaining a lightweight distilled student network.

In this paper, we studied oriented object detection in remote sensing images and model compression for object detection, and have made some progress, but there are still some shortcomings. Therefore, we put forward some research directions in the future:

1. In model compression, we just distilled the prediction components of the network, but the imbalance between positive and negative samples was not considered. For the student network to better learn positive samples and suppress learning background pixels, attention-guided knowledge distillation can be researched in future research work;

2. The relationship between different objects contains valuable information. There are some works such as non-local [40], which can make the detector capture and use this information well. However, the existing knowledge distillation method for object detection in remote sensing images only extracts the information of single pixels, ignoring the relationship between different pixels. Therefore, we can study non-local knowledge distillation to help the student network learn the relationship between different pixels of the teacher network in the future.

5. Conclusions

In this paper, we propose a lightweight based on the keypoints of multi-category detector designed for arbitrary-oriented objects in remote sensing images. Considering that the complex background in remote sensing images will lead to rough feature maps, we proposed a semantic transfer block (STB) to provide a more effective feature fusion method. This is a module that uses semantic information as a guide to align the boundary information of low-level features to refine features. Due to the variety of object sizes in remote sensing images, an adaptive Gaussian kernel (AGK) was proposed to adapt to objects of different scales, eliminate the influence of fuzzy samples in the training stage, and further improve detection performance. It is worth mentioning that BKDet is also an anchor-free detector, which avoids some of the shortcomings of anchor-based methods. Finally, we use knowledge distillation to learn a lighter remote sensing image object detector. We propose the corresponding distillation loss and use the complex detector as a teacher network to guide the student model learning. It is worth noting that the experimental results show that the parameter of the student network obtained by our distillation framework is significantly less than that of the teacher network, but its performance is close to the teacher network, which shows the validity of the method in this paper.

We conduct experiments on the HRSC2016 and UCAS-AOD datasets. The results show that the teacher network can achieve better performance in oriented object detection for remote sensing images. The student network obtained by knowledge distillation can greatly reduce the model parameters and calculation complexity with a small loss of accuracy, which is very valuable for practical applications.

Author Contributions: Methodology, H.M., X.P.; data processing and experimental results analysis, H.M., R.L., X.P. and Y.L.; oversaw the experiments and made suggestions, Y.L. and L.J.; writing—review and editing, Y.L. and H.M.; investigation, H.M., R.L., X.P., Y.L. and R.S.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 61772399, Grant U1701267, Grant 61773304, Grant 61672405 and Grant 61772400, the Key
Research and Development Program in Shaanxi Province of China under Grant 2019ZDLGY09-05, the Program for Cheung Kong Scholars and Innovative Research Team in University Grant IRT_15R53, and the Technology Foundation for Selected Overseas Chinese Scholar in Shaanxi under Grant 2017021 and Grant 2018021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.


Acknowledgments: The authors would also like to express their sincere thanks to the editors and the anonymous reviewers for their valuable comments and contributions.

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

References


