remote sensing

MDPI

Article

An Aircraft Detection Framework Based on Reinforcement Learning and Convolutional Neural Networks in Remote Sensing Images

Yang Li [1,2], Kun Fu [1,2,*], Hao Sun [1] and Xian Sun [1]

[1] Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China; liyang415@mails.ucas.ac.cn (Y.L.); sun.010@163.com (H.S.); sunxian@mail.ie.ac.cn (X.S.)

[2] University of Chinese Academy of Sciences, Beijing 100190, China

* Correspondence: fukun@mail.ie.ac.cn; Tel.: +86-10-5888-7208 (ext. 8931)

Received: 21 December 2017; Accepted: 2 February 2018; Published: 6 February 2018

Abstract: Aircraft detection has attracted increasing attention in the field of remote sensing image analysis. Complex background, illumination change and variations of aircraft kind and size in remote sensing images make the task challenging. In our work, we propose an effective aircraft detection framework based on reinforcement learning and a convolutional neural network (CNN) model. Aircraft in remote sensing images can be accurately and robustly located with the help of the searching mechanism that the candidate region is dynamically reduced to the correct location of aircraft, which is implemented through reinforcement learning. The detection framework overcomes the difficulties that the current detection methods based on reinforcement learning are only able to detect a fixed number of objects. Specifically, we adopt the restricted EdgeBoxes that generate the high-quality candidate boxes through the prior aircraft knowledge at first. Then, we train an intelligent detection agent through reinforcement learning and apprenticeship learning. The detection agent accurately locates the aircraft in the candidate boxes within several actions, and it even performs better than the greed strategy in apprenticeship learning. During the final detection step, we carefully design the CNN model that predicts the probability that the localization result generated by the detection agent is an aircraft. Comparative experiments demonstrate the accuracy and efficiency of our aircraft detection framework.

Keywords: aircraft detection; reinforcement learning; apprenticeship learning; convolutional neural network

1. Introduction

With the development of remote sensing technology and the improvement of image resolution, the automatic aircraft detection in high-resolution remote sensing images not only plays an important role in military application, but also becomes a hot spot in civil aviation field. Aircraft detection is one of the significant research areas in remote sensing image analysis.

Recently, object detection methods [1–4] using convolutional neural networks have been proposed. With the help of these state-of-art methods, aircraft detection in remote sensing images has been greatly developed. The works in [5,6] benefited from the rich feature representation of CNN model and effectively detected aircraft in remote sensing images. They outputted the coordinates of aircraft through the regression of neural networks.

The works in [7–9] took the object detection problem as the Markov Decision Process and trained detection agent based on reinforcement learning. The detection agent adopts the top-down searching process that firstly analyses the global image and then narrows down the local regions that contain

object information step by step. However, these detection methods based on reinforcement learning only detect fixed number of objects, which can not address the challenge of aircraft detection in remote sensing images.

In our work, we propose an effective aircraft detection framework based on reinforcement learning and CNN (RL-CNN) model, which is shown in Figure 1. The process of localization aircraft can be seen as an action decision problem with a sequence of actions to refine the size and position of bounding box. Active interaction to understand the image region, change of the correct bounding box aspect radio and selection region of interest are important to determine the accurate position of aircraft. Based on the characteristics of reinforcement learning and our specific aircraft localization process, we use reinforcement learning to learn and implement our framework. Compared with the object detection method based on reinforcement learning, our aircraft detection framework combines the advantages of reinforcement learning and supervised learning, and is able to detect unfixed number of aircraft in remote sensing images. In addition, compared with the structured prediction bounding box regression algorithms, our detection agent dynamically localizes aircraft.

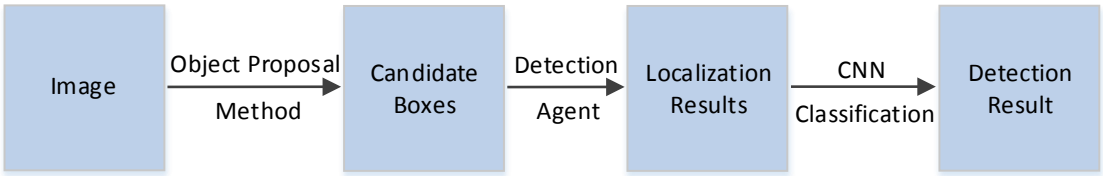


Figure 1. Reinforcement learning and CNN (RL-CNN) aircraft detection framework.

In our detection framework, an object proposal method is used to generate candidate boxes from original image. The detection agent training by reinforcement learning locates the aircraft in candidate boxes. Then, the candidate boxes converge around the aircraft. The CNN model scores each localization result, and we can detect any number and kind of aircraft in remote sensing images after non-maximum suppression.

Our work has the following contributions:

- We firstly combine the reinforcement learning and supervised learning in our aircraft detection framework. We train the aircraft detection agent with the deep Q learning method, and train the CNN model with supervised learning to learn the appearance characteristics of the aircraft.
- We train the detection agent with reinforcement learning and apprenticeship learning, which guide the detection agent with the greed strategy. The detection agent even performs better than the greed strategy in some test samples.
- Our detection framework overcomes the difficulty that the current detection method based on reinforcement learning can only detect a fixed number of objects. We can detect any number and kind of aircraft in remote sensing images.

2. Related Work

2.1. Aircraft Detection Method

Since many works have been done in aircraft detection of remote sensing images over the past years, it is still hard to use a universal detection framework to overcome all challenges in complex remote sensing images.

Sun et al. [10] adopted the spatial sparse coding bag-of-word model to detect aircraft. Zhang et al. [11] proposed a rotation invariant parts based model to improve the detection performance. Liu et al. [12] adopted the coarse-to-fine shape approach to recognize aircraft. These above approaches are effective in particular area. However, they often adopt the sliding-window, which followed a fixed

path to search aircraft with redundant calculation. On the other hand, they have poor generalization because of the manual designed features.

Benefitting from the neural network, the detection frameworks based on rich feature representation and end-to-end training perform well. Diao et al. [13] built a deep belief net (DBN) model pre-trained by the restricted Boltzmann machines (RBM) to detect aircraft. Chen et al. [14] combined a deep convolutional neural network (DNN) with a deep belief net (DBN) in the task of aircraft detection. Xu et al. [15] proposed a unified aircraft detection framework to predict aircraft bounding boxes and class probabilities directly from an arbitrary-sized remote sensing image with the help of the fully convolutional network (FCN). Works in Wang et al. [16], Xu et al. [17] and Zhao et al. [18] adopted the convolutional neural network in the processing of SAR images. Wu et al. [5] combined the object proposal method and convolutional neural network (CNN). With the deep neural network, Zhang et al. [6] proposed a multi-model ensemble method, which decreased the false alarm rate without prior information of interest regions in airport.

2.2. Deep Reinforcement Learning

Reinforcement learning is aimed at learning with rewards and sequential decision-making. It has been widely used in various applications such as robotics control, medical treatment, finance and games. For the first time, the work in DeepMind [19] proposed the combination of reinforcement learning and deep neural network to play Atari 2600 video games. In some games, the agent trained with q learning achieves superhuman performance. In addition, the AlphaGo in [20] won the Go competition studied by professional players for thousands of years. The agent in [21] trained through apprenticeship learning algorithms can directly control autonomous helicopters. The recent studies on reinforcement learning research focus on discrete or continuous control, value function [22] or policy [23]. Different from the methods with supervised learning, the approaches based on reinforcement learning attract more and more attention. The reinforcement learning proves a new method to handle traditional computer vision problems, such as visual tracking [24], action recognition [25] and object detection [7–9], by employing the deep reinforcement learning algorithms.

In the past two years, some object detection methods based on reinforcement learning have been proposed. With deep reinforcement learning, the agent in [7] learned a policy to locate a fixed number of objects through gradually narrowing the bounding box from the hole image to the ground truth. Actions in [7] can change the scale, location or aspect ratio of bounding boxes. The work in [8] reduced the number of actions to six, and it made the policy easier to optimize. Like [7], the detection framework in [8] also used the inhibition-of-return mechanism to locate a fixed number of objects. Different from [7], the agent in [8] adopted the hierarchical representation, which preformed the top-down search to locate objects. The work in [9] proposed a new method based on reinforcement learning to efficiently generate object proposals. The agent in [9] balanced the localization of covered objects and the exploration of uncovered ones with an effective reward function.

3. Proposed Detection Framework

In this section, we present the details of our aircraft detection framework (RL-CNN). In our work, the localization process is defined as the sequential Markov Decision Process (MDP) instead of the regression of the other detection framework based on neural network. With reinforcement learning, we train a detection agent that sequentially interacts with the remote sensing images step by step. The agent makes decisions about which area of the image should be concentrated on each step. Finally, the aircraft can be located by the agent.

Figure 1 shows the proposed detection framework. Our aircraft detection framework has three components. Firstly, we adopt the restricted EdgeBoxes that generates the high-quality candidate boxes of the original image at first. Secondly, we train a detection agent through reinforcement learning and apprenticeship learning. The detection agent accurately locates the aircraft in the candidate boxes within seven actions, and it even performs better than the greed strategy in apprenticeship learning.

Finally, we carefully designed the CNN model that predicts the probability that the localization result generated by the detection agent is an aircraft, and we can generate the detection result through the non-maximum suppression. Three parts of the detection framework are detailed in the following sections.

3.1. Object Proposal Method

In the past, sliding windows was the main way to generate candidate boxes. However, it is an exhaustive search over the full image, and it is inefficient and time-consuming. Many object proposal methods were proposed and compared in the past few years [26], such as Bing [27], SelectiveSearch [28], and EdgeBoxes [29]. Since the EdgeBoxes is efficient, we use EdgeBoxes to generate candidate boxes in our work. With the prior aircraft knowledge, we adopt a restricted EdgeBoxes to reduce the number of candidate boxes for the aircraft detection task in remote sensing images. Based on the fixed range size of aircraft, we added constraints to EdgeBoxes. The size of candidate boxes should observe the following rules:

$$W_{min} < w < W_{max}, \tag{1}$$

$$H_{min} < h < H_{max}, \tag{2}$$

$$r = max(w,h)/min(w,h), \tag{3}$$

$$R_{min} < r < R_{max}, \tag{4}$$

where the W_{min}, W_{max}, H_{min}, H_{max}, R_{min} and R_{max} are constant value, and w and h are the width and height of candidate box. Thus, the widths, heights and their ratios of candidate boxes are in a specific range. Restricted EdgeBoxes can generate a moderate number of high quality candidate boxes of remote sensing images.

3.2. Detection Model Based on Reinforcement Learning

In this section, we present the details of our aircraft detection agent based on deep reinforcement learning. In our work, the aircraft detection is considered as a Markov Decision Process. With reinforcement learning, we train a detection agent, which sequentially interacts with the remote sensing images by selecting actions from the predefined action set. The total cumulative discounted reward represents the accuracy of localization in our aircraft detection task. The goal of detection agent is to maximize the total cumulative discounted reward.

3.2.1. Markov Decision Process in Aircraft Detection

Markov Decision Process (MDP) in our work is that the agent periodically and continuously observes the Markov Dynamic System (environment) and makes a sequential decision. MDP consists of three parts, the set of action A, the set of state S, and the reward function R. The agent receives the state s from the environment; then, it makes the appropriate action a on the environment. The environment receives the action, it transfers into the new state $s^{'}$ and generates reward r of next time step. This is the basic transition $(s, a, r, s^{'})$ in MDP of aircraft detection. Three components of the MDP in our detection framework are detailed in the following parts.

Action: The action set contains six actions. According to the results after choosing an action, the actions can be divided into two categories: one kind of action reduces the size of the image window to get a new observation area, the other kind of action indicates the aircraft is located and the MDP process ends. The reducing size actions contain five actions: top-left (A1), top-right (A2), bottom-left (A3), bottom-right (A4) and middle (A5). After choosing this kind of action, we make the width and height of new region retain three quarters of the upper image, and the shrink rate of the region is set to 3/4. When the shrink rate of the region is big, the new region retains more content of the upper image, and the localization process becomes longer. When the shrink rate of the region is small, the new region retains less information of the upper image, and the accuracy of localization decreases. Thus,

we set the shrink rate to 0.75 for balancing the accuracy and efficiency of localization. The end action (A6) retains the entire upper image and takes it as the localization result. With this action set, we can balance the accuracy and speed of aircraft localization. Figure 2 shows the action set in MDP, and the new image windows are surrounded by red boxes after selecting actions.

State: Instead of the state in [9], in our work, the state of MDP is a combination of two parts: the current region of image that indicates what the detection agent sees, and the history actions that the agent selected. These kinds of history actions are also used in [8] instead of the past frames. The history of selected actions is made up of the latest five one-hot vectors in series according to the chronological order, and each one-hot vector indicates which action is selected. Since there are six actions in the action set, the history action vector has 30 dimensions. Like the history action in [7], the history actions in our work contribute to the stability of localization.

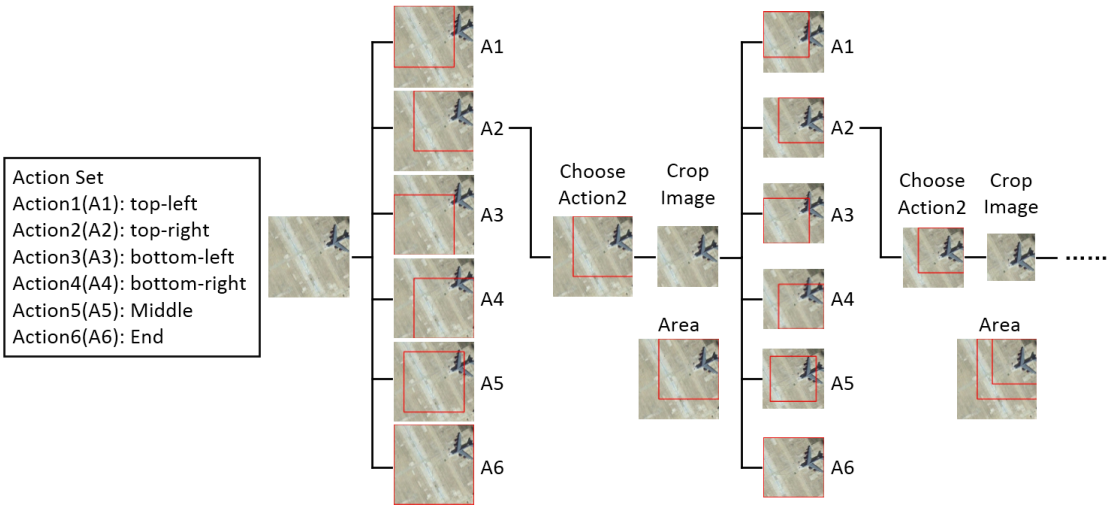


Figure 2. The action set and search process of the detection agent.

Reward: The reward function plays an important role in MDP. The reward function $R_a(s_t, s_{t+1})$ stand for that in state s_t, the agent selects action a, and the state transfers to s_{t+1}; then, the environment will give the reward to encourage or punish the detection agent according to the reward function. In our aircraft detection framework, the reward function indicates whether the action selected by the detection agent is good or not. We use the Intersection-over-Union (IoU) between the current region and the ground truth of aircraft to construct the reward function. The IoU between current region box and ground truth box is defined as:

$$IoU(r, g) = area(r \cap g)/area(r \cup g), \tag{5}$$

where the r stands for the current region box, and the g stands for the ground truth box.

For the reducing size actions, the reward function returns the improvement of aircraft localization accuracy. The reward function of this kind of action is defined as:

$$R_r(s_t, s_{t+1}) = sign(IoU(r_{t+1}, g) - IoU(r_t, g)), \tag{6}$$

where, at time t, the region is r_t, the state is s_t, and the Intersection-over-Union is $IoU(r_t, g)$. Then, the agent takes action a_t, the region r_t transfers into r_{t+1}, the state transfers into s_{t+1}, and the Intersection-over-Union changes to $IoU(r_{t+1}, g)$. When the IoU is increased, the agent gets a positive $+1$ reward. Otherwise, the agent gets a negative -1 reward.

For the end action, the MDP is ended, and the reward function returns the aircraft localization accuracy. The reward function of this kind of action is defined as:

$$R_e(s_t, s_{t+1}) = \begin{cases} +\eta, & \text{if } IoU(r_{t+1}, g) \geq \tau, \\ -\eta, & \text{otherwise,} \end{cases} \tag{7}$$

where η is the end reward, and we set η to 3.0 in our work. If the IoU between the final region and the ground truth box is greater than the fixed threshold τ, the end reward is +3, and -3 otherwise. In the traditional aircraft detection, the threshold τ is often set to 0.5. This means that, when the final IoU is greater than 0.5, the detection result is considered as an aircraft. When the terminative threshold τ is big, the detection agent needs to search for more steps to achieve high IoU. When the τ is small, the termination condition is easy for the detection agent so the accuracy decreases. Finally, the τ is set to 0.6 for the balance between the accuracy and speed of aircraft localization.

3.2.2. Apprenticeship Learning in Aircraft Detection

The learning process of humans or animals are usually combined with imitation. According to the imitation, learners can avoid global searches and they can focus on more significant local optimizations. The kind of learning form in reinforcement learning based on expert demonstrations is called apprenticeship learning. Apprenticeship learning leverages human knowledge to efficiently learn good controllers.

In our work, we exploit prior human knowledge to help the detection agent learn the action control. We adopt the following prior greedy strategy:

Since we know the ground truth boxes in the training state, we can get six IoUs for six actions before the agent selects action. According to different IoUs, we can calculate different positive and negative rewards for each action. Thus, we guide the agent to take the action that can achieve the highest IoU. The action choosing based on greedy strategy observes the following rules:

$$IoU_{next} = \{IoU_{A1}, IoU_{A2}, IoU_{A3}, IoU_{A4}, IoU_{A5}, IoU_{A6}\}, \tag{8}$$

$$a_{next} = argmax(IoU_{next}), \tag{9}$$

where IoU_{next} contains six different IoUs before selecting action. In apprenticeship learning, we guide the agent to choose the action a_{next} for the highest IoU.

3.2.3. Deep Q Network Optimization

The detection agent interacts with the remote sensing images to maximize the cumulative discounted reward. We make a discount to future rewards through the factor γ; in our work, we set γ to 0.9. At time step t, the cumulative discounted reward (CDR) is defined as:

$$CDR_t = \sum_{t'=t}^{T} \gamma^{t'-t} R_{t'}, \tag{10}$$

where T is the moment that the searching process is ended, and $R_{t'}$ is the reward the agent receives, which is defined in Section 3.2.1.

At state s, the action value function $Q(s, a)$ guides the detection agent to select action a. Furthermore, according to Bellman equation, we can iteratively update to estimate the $Q(s, a)$. The neural network with weights θ is used for approximating the action value function $Q(s, a)$.

Instead of the single Q model in [7,9], we use the Q model with the target $\hat{Q}$ model in [30]. When we process the optimization, the parameters of the previous iteration θ^- remain unchanged

through the target $\hat{Q}$ model. The target network makes the optimization more stable. The update of neural network weights is defined as:

$$\theta_{i+1} = \theta_i + \beta(R + \gamma max_{a'}\hat{Q}(s', a'; \theta_i^-) - Q(s, a; \theta_i))\nabla_{\theta_i}Q(s, a; \theta_i), \quad (11)$$

where a' is the action the detection agent can select at state s', β represents the learning rate and γ represents the discount factor.

The pseudo code for training the deep Q network is shown in Algorithm 1. Based on the $\varepsilon - greedy$ policy and apprenticeship learning, the detection agent selects action to explore the new state and exploit the experience.

Algorithm 1 Deep Q Learning with Apprenticeship Learning.

Initialize experience buffer deque E
Initialize $\varepsilon - greedy$ policy with $\varepsilon = 1.0$
Initialize Deep Q Network Q with random weights θ
Initialize target Deep Q Network $\hat{Q}$ with weights $\theta^- = \theta$
for epoch = 1, M **do**
  $\varepsilon = \varepsilon - 0.1$ until $\varepsilon <= 0.1$
  Update $\theta^- = \theta$ each L training steps
  for image-number = 1, N **do**
    Initialize image window w_1, history action h_1 and terminative flag $f = False$
    Construct initial state $s_1 = (w_1, h_1)$
    while $f == False$ **do**
      if random < ε **then**
        select a random action a_t from $\{A1, A2, A3, A4, A5, A6\}$
      else
        $a_t = argmax(IoU_{next})$
      Perform a_t and get new window w_{t+1}, history action h_{t+1} and reward r_t from environment
      Construct new state $s_{t+1} = (w_{t+1}, h_{t+1})$ and get new MDP unit (s_t, a_t, r_t, s_{t+1})
      Push MDP unit (s_t, a_t, r_t, s_{t+1}) into deque E
      Update state $s_t = s_{t+1}$
      if $a_t == A6$ **then**
        $f = True$
      else
        $f = False$
      Sample random batch of units (s_j, a_j, r_j, s_{j+1}) from E
      if $a_j == A6$ **then**
        $y_j = r_j$
      else
        $y_j = r_j + \gamma max_{a'}\hat{Q}(s_{j+1}, a'; \theta_i^-)$
      Update the network parameters θ using backpropagation of $(y_j - Q(s_j, a_j; \theta))^2$

3.2.4. Deep Q Learning Model

The MobileNets in [31] greatly optimizes the speed and model size, and it maintains state-of-art accuracy with the Depthwise Separable Convolution unit. In the stage of image feature extraction, we adopt the pre-trained MobileNets to generate features of the image. As illustrated in Figure 3, our Q network contains four fully connected layer.

The first fully connected layer is the connection of two components: the 30-d history action and the 50176-d image features generated from the last convolutional layer of MobileNets. The second

fully connected layer is combined with the Rectified Linear Unit (ReLU) and Dropout layers, and has 4096 neuron nodes. Then, the third fully connected layer is also combined with the ReLU and Dropout layers, and has 1024 neuron nodes. Finally, the last fully connected layer represents the q value of the action, and it has six output nodes in our work.

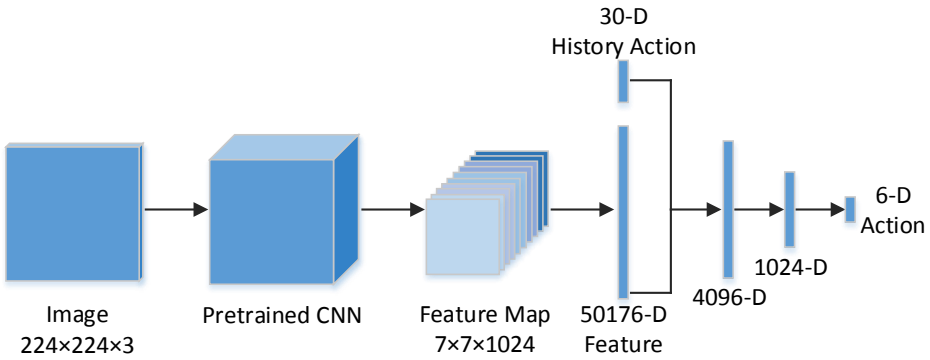


Figure 3. The Deep Reinforcement Learning Detection model.

3.3. Convolutional Neural Network Model

Presented by the Visual Geometry Group of the Oxford University, VGGNet in [32] is No.2 in classification task of the ILSVRC-2014. It achieves outstanding contribution through adopting the small convolution (3×3) and increasing the depth of network, and VGGNet has a perfect generalization ability for other datasets.

Due to the advantage of VGGNet, we use the pre-trained VGGNet in our CNN model to classify the image region located by the detection agent. We also adopt the BatchNormalization and Dropout layer in our CNN classification model. The BatchNormalization layer addresses the gradient problem of backpropagation and improves the capacity of the classification model. The Dropout layer prevents over-fitting and enhances the classification model.

We adopt a fully connected layer with 4096 neuron nodes at the top of the last convolutional layer in VGGNet, which extracts feature vectors to describe the image region. After the first fully connected layer, we employ a fully connected layer with 1024 neuron nodes. Finally, in the last fully connected layer, we use two neuron nodes instead of one thousand nodes in original VGGNet to solve the two category classification problem whether the image region is an aircraft or background. The CNN model outputs the probability that the image region is an aircraft.

In the detection part, we adopt the non-maximum suppression to reduce the number of extra bounding boxes for each aircraft. Then, we can get the final detection result.

4. Experiments and Analysis

We design several experiments to evaluate the performance of our aircraft detection framework in this section. Our detection framework consists of three parts: firstly, the restricted EdgeBoxes generates the candidate boxes of the original image; secondly, the detection agent locates the aircraft in the candidate boxes; and, finally, the CNN model predicts the probability whether the location result is an aircraft, and we can get the final detection results through Non Maximum Suppression (NMS). We expand the dataset in [5] to generate our own training and testing samples, which are collected from the Google Earth through the QuickBird. The ground truth bounding box is the minimum bounding rectangle of the aircraft, and all ground truth bounding boxes are manually annotated. We use our dataset to train and test the detection agent and the CNN model.

Our reinforcement learning training dataset contains 2700 RGB images, and the testing dataset contains 500 RGB images with the same size of training data. The training and validation dataset of our CNN model contains 30,000 positive and 30,000 negative RGB sub-images cropped from the original images through the restricted EdgeBoxes. Finally, we test our aircraft detection system on the final 20 remote sensing images. All experiments are done on one NVIDIA K40M 12G GPU. Our detection framework is implemented using Keras.

4.1. Evaluating the Performance of Restricted EdgeBoxes

In this section, we compare the performance of the restricted EdgeBoxes, initial EdgeBoxes [29] and CEdgeBoxes [6]. H_{min} and W_{min} are set to 20, H_{max} and W_{max} are set to 160, R_{min} is set to 1.0 and R_{max} is set to 2.0. In traditional detection tasks, an object is detected if the IoU between the detection bounding box and the ground truth box is more than 0.5.

When we change the threshold of the IoU from 0.0 to 1.0, we can get different recall rates (RR) of aircraft. After drawing the RR-IoU curve, we can calculate the area between the RR-IoU curve and the IoU axis.

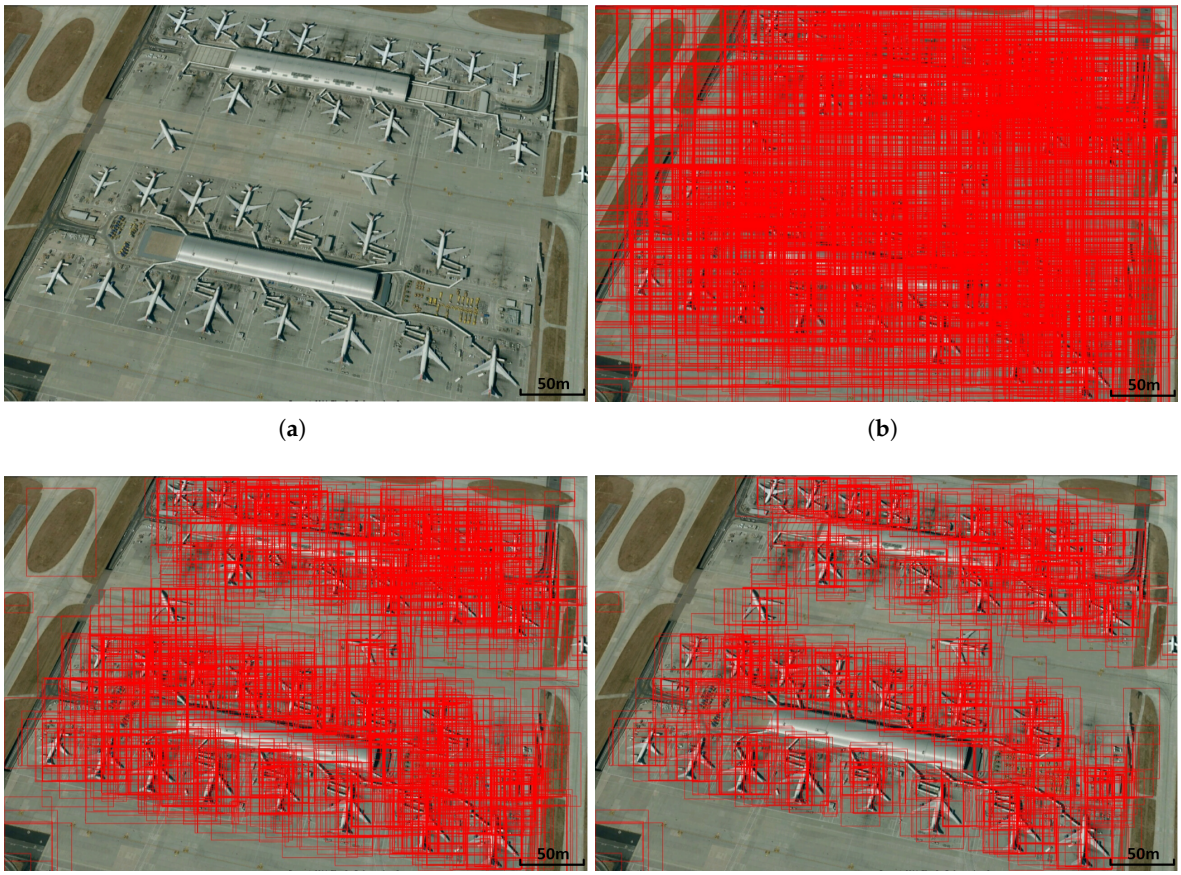
- We use the average recall rate of aircraft (ARR) to stand for the value of this area. The ARR reflects the extent of coverage between the candidate bounding boxes and the ground truth.
- We use the middle recall rate (MRR) of the aircraft stand for the recall rate, for which the threshold of IoU is set to 0.5. The MRR reflects the localization accuracy.

In the condition that the maximal number (MN) of candidate boxes per image is set to 5000, we use average number of candidate boxes per image (AN), average recall rate of aircraft (ARR), middle recall rate of aircraft (MRR) and average time per testing image (ATPI) as the evaluation standard to compare EdgeBoxes, CEdgeBoxes and the restricted EdgeBoxes.

Table 1 shows that the restricted EdgeBoxes can get a compromise effect between the number of region proposals and the recall rate of aircraft. EdgeBoxes gets the 96.01% MRR and 77.05% ARR with the average 5000 candidates per remote sensing image. CEdgeBoxes gets the 96.01% MRR and 77.01% ARR with the average 3704 candidates per remote sensing image. In addition, the restricted EdgeBoxes gets the 96.01% MRR and 76.96% ARR with the only average 2386 candidates per remote sensing image. With the decrease of candidates, the restricted EdgeBoxes need less time to write candidates and labels to disks compared with other methods, about 0.516 s per testing image. Compared with EdgeBoxes, the restricted EdgeBoxes almost achieves the same average recall rate performance with only half of the candidates, and its processing speed is faster. Figure 4 shows the results of different methods that generate candidate boxes. The restricted EdgeBoxes significantly reduce the number of candidate boxes.

Table 1. Performance of different object proposal methods.

Object Proposal Method	MN	AN	ARR	MRR	ATPI(s)
Restricted EdgeBoxes	5000	2386	76.95%	96.01%	0.516
CEdgeBoxes [6]	5000	3704	77.01%	96.01%	0.712
EdgeBoxes [29]	5000	5000	77.05%	96.01%	0.947



(**a**) (**b**)

(**c**) (**d**)

Figure 4. Display of the results through different methods that extract candidate boxes. All results are located with red bounding boxes. (**a**) the original remote sensing image; (**b**) the results generated by EdgeBoxes [29]; (**c**) the results generated by CEdgeBoxes [6]; and (**d**) the results generated by the restricted EdgeBoxes.

4.2. The Detection Agent with Apprenticeship Learning

4.2.1. Training of Detection Agent

Through reinforcement learning, we train our detection agent on 2700 images. The parameters of the CNN feature part are fixed during the training. We make the detection agent interact with the 2700 images. When the detection agent finished interacting with all of the 2700 remote sensing images, one training epoch is finished, and we train the agent for 30 epochs. We use the $\varepsilon - greedy$ policy in the training step. With the training going, the value of ε decreases from 0.9 to 0.1, which means that, in the beginning of the training, the detection agent has larger probability 0.9 to selection actions randomly for exploring the different states. The value of ε reduces 0.1 when one epoch is finished, and it is fixed to 0.1 at last. In the later epochs, the detection agent is more likely to select actions based on the trial and error for exploiting the experience learned by itself.

We train two kinds of detection agents. The first agent selects random actions in the exploration stage, and we call it the without-knowledge agent. On the other hand, we know the ground truth of the aircraft in the training stage, so the second agent selects actions guided by the greed strategy in apprenticeship learning that teaches the agent whose action is the best to get the largest increase of the IoU. We call it the with-knowledge agent. Each kind of the detection agent totally takes 120 million actions in 30 epochs, and the deep Q network updates 120 million times. The detection agent takes five actions to locate the aircraft on average.

Following the pseudo code in Algorithm 1, we train the deep Q network. There are some differences between Algorithm 1 and the algorithm in [30].

- The algorithm in [30] guides the detection agent to select actions based on the deep Q network. Different from the algorithm in [30], Algorithm 1 adopts apprenticeship learning to guide the detection agent at each training step.
- Algorithm 1 makes the ε in $\varepsilon - greedy$ decrease when each training epoch is finished. However, the algorithm in [30] lets ε decrease at each training step. With this method, the training process is more stable.
- The algorithm in [30] trains the deep Q network to play an Atari 2600 video game. However, the remote sensing image is more complex compared with the simple game screen. We set bigger experience replay buffer, and the buffer can store the MDP transitions of one training epoch.

4.2.2. Testing of Detection Agent

In this section, we evaluate the performance of our detection agents based on reinforcement learning and apprenticeship learning. We test our detection agents on 500 images. Since our agents directly give the locations of the aircraft in the images without outputting the confidence of the localization results, we do not use the traditional evaluation criteria on the test images. We continue to use the ARR and the MRR to evaluate the performance. We evaluate the random action agent, without-knowledge agent and with-knowledge agent in the testing 500 remote sensing images. The random action agent randomly selects actions without the guidance of the deep Q network.

Figure 5 shows how the performance of the agents improves at different epochs on the testing 500 images. As Figure 5a,b showed, the with-knowledge agent performs better than the without-knowledge agent and random action agent. After 30 epochs, the with-knowledge agent achieves 0.56 ARR and 0.76 MRR, the without-knowledge agent achieves 0.48 ARR and 0.65 MRR, and the random action agent that randomly chooses actions only achieves 0.05 ARR and 0.05 MRR.

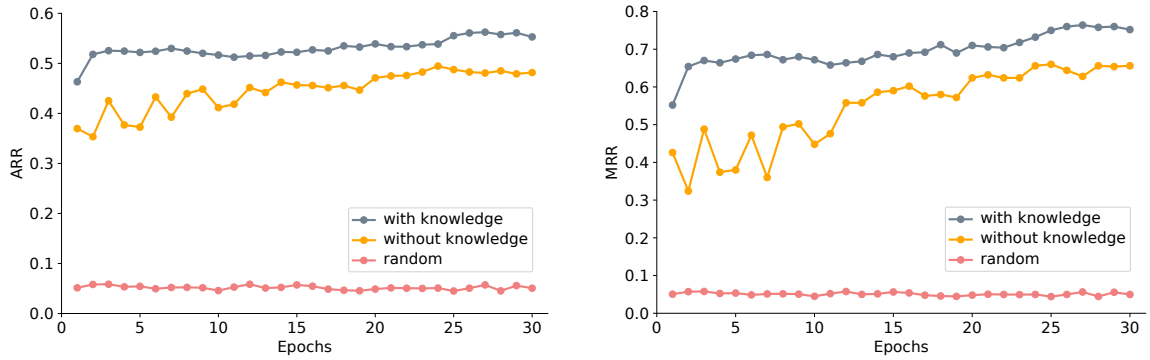


Figure 5. Performance of the random action agent, without-knowledge agent and with-knowledge agent at different epochs. (**a**) performance of the average recall rate (ARR) at different epochs; and (**b**) the middle recall rate (MRR) at different epochs.

We also test the detection agent on the candidate boxes that are generated by the restricted EdgeBoxes of the final 20 remote sensing images after 30 epochs. Table 2 shows the ARR and MRR criteria of aircraft for different detection agents. The with-knowledge agent achieves the best performance. As Tables 1 and 2 show, the MRR of testing image is improved to 99.34% contrasted with 96.01% in Table 1 after the processing of with-knowledge detection agent.

Table 2. Performance of different detection agents.

Detection Agent	Average Recall Rate (ARR)	Middle Recall Rate (MRR)
Random Action Agent	61.08%	79.07%
Without-Knowledge Agent	71.85%	91.03%
With-Knowledge Agent	77.25%	99.34%

According to the test results, we take with-knowledge agent as the final detection agent, which is used in the following experiments.

Sometimes, the policy based on the detection agent, which learns through the reinforcement learning and apprenticeship learning, can even perform better than the greedy strategy in apprenticeship learning. Figure 6 shows that, in the same testing image, the greedy strategy and the detection agent take different actions, and they get different IoUs and localization results. The actions based on the greedy strategy follow the rule: taking the action that can get the highest IoU in each decision-making process. At last, the greedy strategy gets the 0.73 IoU for the test image. However, the detection agent trained by the greedy strategy achieves the 0.92 IoU for the same image. In the whole searching process, the agent does not take the action that achieves the highest IoU in the beginning, so the agent achieves a smaller IoU. However, in the long run, the agent learns to abandon the immediate interests and focus on the long-term reward. The detection agent is able to avoid the local optimization.

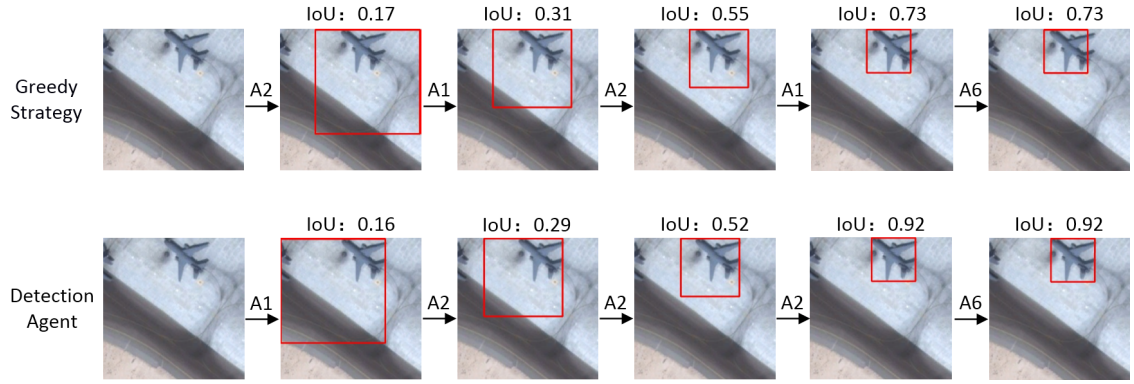
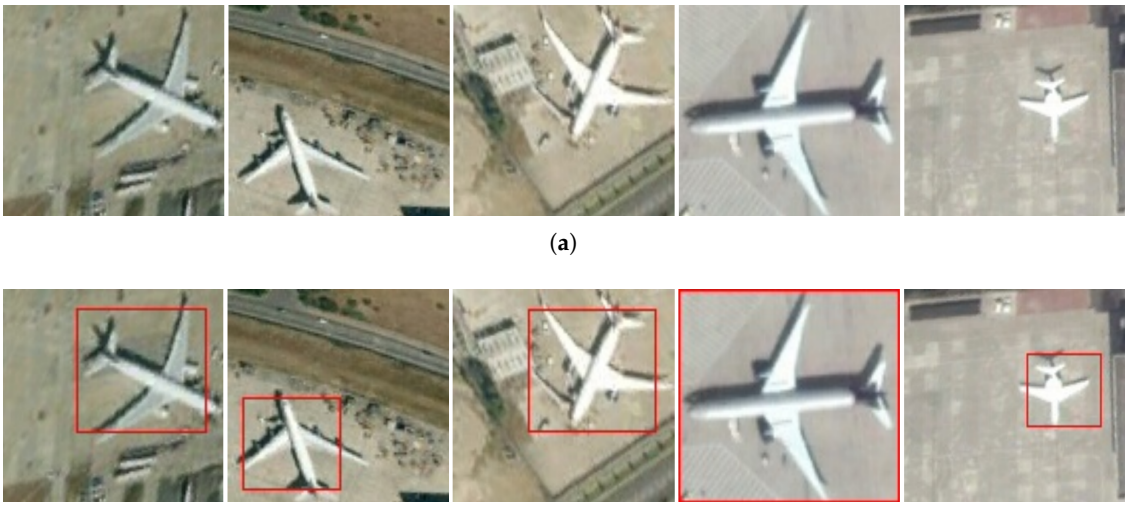


Figure 6. The first row is the action sequence and IoU based on the greedy strategy in apprenticeship learning, which is taking the action that can get the highest IoU at each time step. The second row is the action sequence and IoUs based on the detection agent, which learn according to the greedy strategy.

After the restricted EdgeBoxes processing, we can get many candidate boxes that may contain the aircraft. For each candidate box, the detection agent takes the whole candidate box and the 30-d action vector as the initial state. Then, the detection agent interacts with the environment, it apperceives the state, and it makes decisions about which action should be taken. Furthermore, the environment receives the new action and gives the new state and reward to the agent. Finally, the agent makes the terminate action; then, the final bounding box is regarded as the localization result.

Figure 7 shows the results that the detection agent localized on the candidate boxes. As the first three examples showed in Figure 7a,b, the three aircraft are in different locations, and the detection agent can successfully locate the aircraft. As the fourth example showed in Figure 7a,b, when the candidate box contains the whole aircraft, the detection agent takes the original candidate box as the localization result. As the last example showed in Figure 7a,b, when the aircraft size is small, the detection agent takes a long search path from the original candidate box and gets the final accurate localization result. Thus, the detection agent has a robust property to locate the different poses, kinds and sizes of the aircraft.

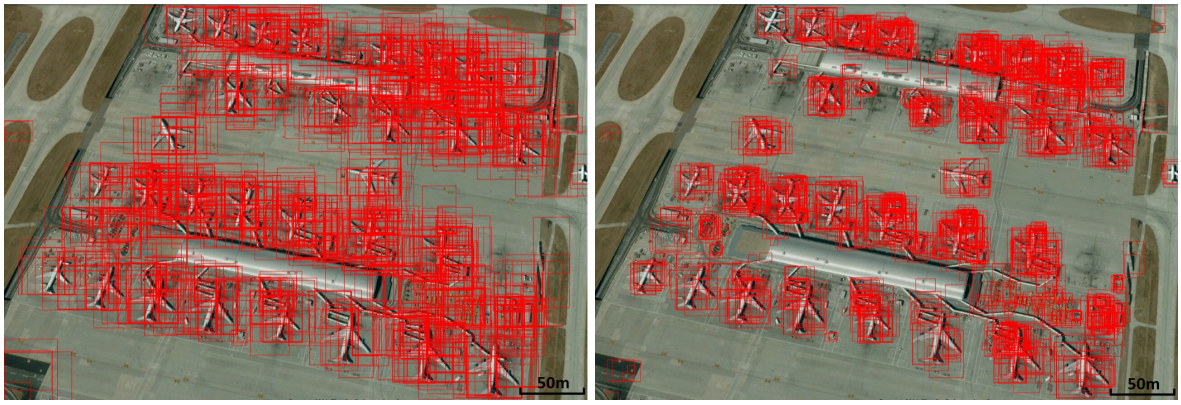
Figure 8b shows the bounding boxes generated by the detection agent in one test image. Compared with Figure 8a, the bounding boxes in Figure 8b converge on the aircraft.



(**a**)

(**b**)

Figure 7. Display of the results that the detection agent localized on the clips generated from final 20 remote sensing images. All results are located with red bounding boxes. (**a**) the clips of remote sensing images generated by the restricted EdgeBoxes; and (**b**) the localization results of the clips through the detection agent.



(**a**) (**b**)

Figure 8. Display of the result that the detection agent generated on one test image which has 1296×883 pixels. All boxes are located with red bounding boxes. (**a**) an example of the candidate boxes generated by the restricted EdgeBoxes on the test image; and (**b**) the result the detection agent generated, and the bounding boxes converge on the aircraft.

4.3. The Unfixed Number of Aircraft Detection in RL-CNN

In this section, we evaluate the unfixed number of aircraft detection in our RL-CNN. We compare our RL-CNN aircraft detection framework with the Hierarchical Object Detection with Deep Reinforcement Learning (HODRL) in [8]. We evaluate both aircraft detection frameworks on the testing remote sensing image.

Like [7], the detection agent in HODRL locates objects from the hole image. To avoid endless attractions towards the region of interest, the detection agent in HODRL also adopts the inhibition-of-return (IOR) mechanism with that the historically and currently attended region is

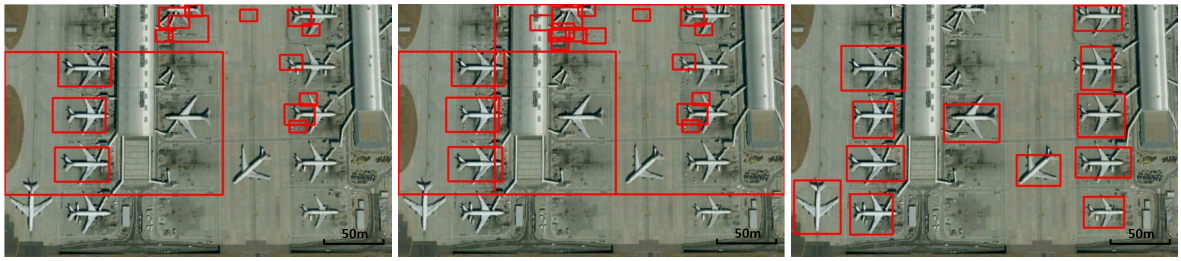
prevented from being attended again. It can locate objects through the fixed number of actions based on prior knowledge in VOC2007 [33] and VOC2012 [34] dataset. We set the largest number of actions made by the detection agent in HODRL to 150 and 200. We can get different detection results through different aircraft detection frameworks.

We calculate the mean IoU which measures the overall degree of closeness between the ground truth and the detection bounding boxes. As Table 3 shows, the mean IoU in RL-CNN is much higher than the mean IoU in HODRL(150) and HODRL(200). Consequently, the bounding boxes generated by RL-CNN locate aircraft accurately.

Table 3. Mean Intersection-over-Union (IoU) of HODRL [8] and RL-CNN detection models.

Detection Model	HODRL(150) [8]	HODRL(200) [8]	RL-CNN
Mean IoU	39.98%	45.44%	69.13%

Figure 9 shows the different detection results through different aircraft detection frameworks. Compared with Figure 9a, there are more detection results in Figure 9b because of the larger number of actions taken by the detection agent in HODRL. As Figure 9a,b shows, the detection agent in HODRL only successfully locates a few aircraft; it also locates the wings and tails of some aircraft; furthermore, for most aircraft, it does not locate any part of them. The detection agent in HODRL has poor performance in remote sensing images because of complex background and the manually designed action number, which hardly handle the large and unknown number of aircraft. Compared with HODRL, Figure 9c shows that our RL-CNN aircraft detection framework, which benefits from the candidate boxes and the detection agent, trained through the reinforcement learning, successfully detects aircraft in the remote sensing image without the prior action number.



(**a**) (**b**) (**c**)

Figure 9. The detection results of different aircraft detection frameworks. (**a**) the detection result in the condition that the largest number of actions in HODRL [8] is set to 150; (**b**) the detection result in the condition that the largest number of actions in HODRL [8] is set to 200; and (**c**) the detection result generated by the RL-CNN detection framework.

4.4. The Overall Detection Performance

In this section, we evaluate the overall performance of our RL-CNN aircraft detection framework based on reinforcement learning and CNN model. An aircraft is detected if the IoU between the detection bounding box and the ground truth is higher than 50%. We adopt the Recall Rate (RR), Precision Rate (PR) and False Alarm Rate (FAR) to evaluate the performance of our detection framework. They are defined as:

$$PR = \frac{TP}{TP + FP}, \tag{12}$$

$$RR = \frac{TP}{TP + FN}, \tag{13}$$

$$FAR = \frac{FP}{TP + FP}. \tag{14}$$

We compare our RL-CNN aircraft detection framework with Histogram of Oriented Gradient Support Vector Machine (HOG-SVM), Multi-model Fast Regions CNN (MFCNN) [6] and Faster Regions CNN (Faster-RCNN) [1], which are effective aircraft detection models. Figure 10 shows Precision–Recall curves of four detection models. RL-CNN detection framework performs better than the other three models.

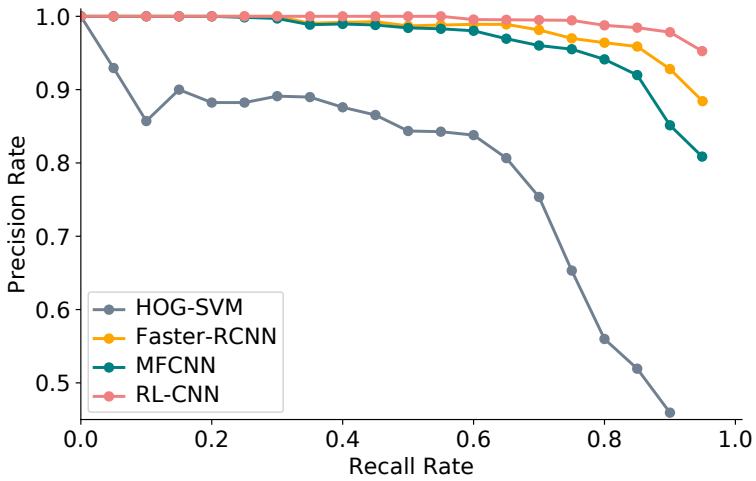


Figure 10. Precision–Recall Curves of HOG-SVM, MFCNN [6], Faster-RCNN [1] and our RL-CNN aircraft detection model. Our RL-CNN framework achieves the best performance.

We calculate the mean IoU, which measures the overall degree of closeness between the ground truth and the detection bounding boxes. As Table 4 shows, the mean IoU generated by Faster-RCNN, MFCNN and RL-CNN is much higher than the mean IoU in HOG-SVM, and our RL-CNN achieves the highest mean IoU. The detection bounding boxes generated by RL-CNN cover the ground truth tightly.

Table 4. Mean IoU of four detection models.

Detection Model	HOG-SVM	MFCNN [6]	Faster-RCNN [1]	RL-CNN
Mean IoU	27.24%	63.39%	67.15%	69.13%

We compare the detection time on each testing image for four aircraft detection frameworks as Table 5 shows. Compared with MFCNN, our RL-CNN generates less candidate boxes. Thus, RL-CNN is faster than MFCNN. Benefitting from the Region Proposal Network, Faster-RCNN can directly process the input remote sensing image, and it needs less time than MFCNN and RL-CNN. Lastly, HOG-SVM only uses the simple HOG feature instead of the neural network, and it costs the least time.

Table 5. Average processing time on each testing image.

Detection Model	HOG-SVM	MFCNN [6]	Faster-RCNN [1]	RL-CNN
Time(s)	1.93	9.22	3.47	5.52

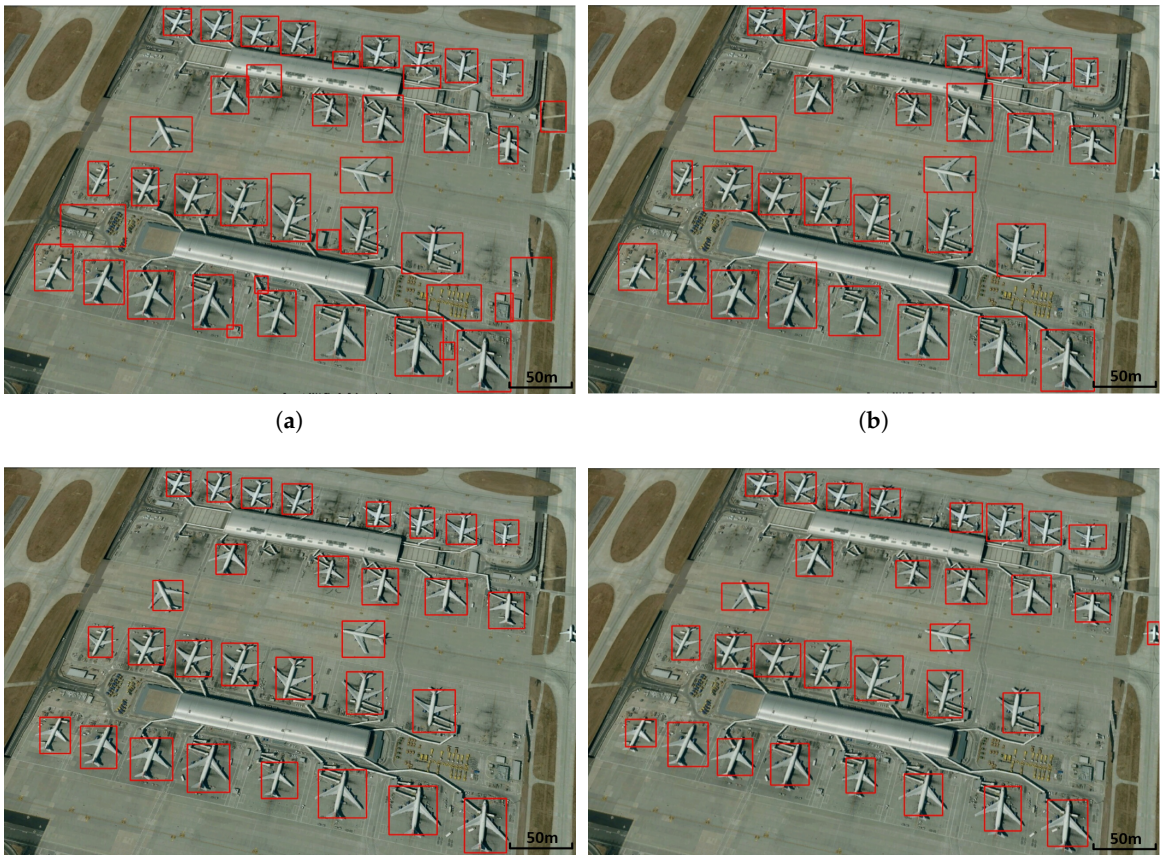
Table 6 shows a False Alarm Rate of four detection models at different given Recall Rates. Since the HOG-SVM uses the artificial HOG feature, sometimes it can not distinguish aircraft from the background in remote sensing images limited by the generalization of models, and it gets the biggest false alarm rate. The biggest RR in HOG-SVM only achieves 89.79%, so HOG-SVM has no

FAR data in 95% RR. Different from the HOG-SVM, MFCNN, Faster-RCNN and RL-CNN benefit from the deep CNN structure, so they can effectively detect aircraft of different scales and shapes. Furthermore, the detection agent in RL-CNN locates aircraft step by step using the top-down searching policy implemented through reinforcement learning, and the RL-CNN finally detects aircraft in candidate boxes. The detection bounding boxes in RL-CNN are tighter compared with the detection bounding boxes in MFCNN and Faster-RCNN. The RL-CNN performs better than HOG-SVM, MFCNN and Faster-RCNN.

Table 6. False Alarm Rate (FAR) of four detection models.

Given Recall Rate	FAR(HOG-SVM)	FAR(MFCNN [6])	FAR(Faster-RCNN [1])	FAR(RL-CNN)
60%	16.20%	1.97%	1.09%	0.44%
65%	19.34%	3.06%	1.11%	0.47%
70%	24.64%	3.98%	1.86%	0.51%
75%	34.68%	4.49%	3.01%	0.55%
80%	44.01%	5.87%	3.60%	1.23%
85%	48.06%	8.01%	4.12%	1.54%
90%	54.09%	14.86%	7.19%	2.17%
95%	–	19.73%	11.57%	4.75%

Some detection results of different methods are shown in Figure 11. The results show that our detection framework can accurately detect aircraft in remote sensing images.



(**a**) (**b**)

(**c**) (**d**)

Figure 11. Detection results of different methods in some test samples. The detection results are located with red bounding boxes. (**a**) the detection results of HOG-SVM; (**b**) the detection results of MFCNN [6]; (**c**) the detection results of Faster-RCNN [1]; and (**d**) the detection results of RL-CNN.

5. Discussion

By analyzing and comparing many groups of experiments, the validity of our proposed aircraft detection framework RL-CNN is verified. The main difference between our RL-CNN and the state-of-the-art aircraft detection framework is that we combine the intelligent detection agent that locates aircraft following the searching policy and CNN classification that is carefully designed to classify aircraft and background.

- We train the aircraft detection agent with a deep Q learning method that is optimized for the aircraft detection task in remote sensing images. With the apprenticeship learning in training, Figure 5 and Table 2 show that the with-knowledge agent achieves better performance compared with the without-knowledge agent, which is trained through the algorithm similar with the work in [30]. More importantly, the with-knowledge agent learns to abandon the immediate interests and focus on the long-term reward to avoid the local optimization, as Figure 6 shows.
- HODRL in [8] is a typical representative of object detection framework based on reinforcement learning. The experiment in Section 4.3 proves that previous detection framework based on reinforcement learning like HODRL is not suited for aircraft detection tasks in remote sensing images. Based on the special combination of detection agent and CNN classification, our RL-CNN is able to accurately locate any number of aircraft in remote sensing images.
- Benefitting from the top-down step by step searching policy that is implemented through reinforcement learning, the detection result can cover aircraft tightly. Consequently, RL-CNN effectively detects aircraft in remote sensing image compared with state-of-the-art aircraft detection frameworks like HOG-SVM, MFCNN [6] and Faster-RCNN [1].

Our RL-CNN aircraft detection framework offers perfect accuracy performance as the Precision–Recall curve and FAR–RR table show. However, the object proposal method still works as one independent part in our RL-CNN. Instead of the independent object proposal method, the region proposal network in Faster-RCNN is specially designed to generate high quality and less candidates. Consequently, the limitation of our work is that RL-CNN costs more running time compared with the state-of-the-art Faster-RCNN detection framework, which is showed in Table 5.

6. Conclusions

In this paper, we firstly propose an effective and novel RL-CNN aircraft detection framework in remote sensing images based on reinforcement learning and the CNN model. The restricted EdgeBoxes generate the high-quality and small number of candidate boxes through the prior knowledge of aircraft. The intelligent detection agent based on reinforcement learns through exploration of new state and exploitation of its own experience, and it can accurately locate the aircraft in the candidate boxes step by step following the top-down searching policy. With the combination of detection agent and CNN models, which predicts the probability that the localization result by the detection agent is an aircraft, our RL-CNN framework can exactly detect aircraft in remote sensing images.

Experiments illustrate that the detection agent in our RL-CNN is able to abandon the immediate interests and focus on the long-term reward to yield superior performance. RL-CNN aircraft detection frameworks can successfully detect unfixed number of aircraft in the remote sensing images without the prior action number. Quantitative analyses of the experimental results demonstrate the state-of-the-art performance of the proposed RL-CNN aircraft detection framework.

Despite the superior performance, there are some shortcomings in our work. The independent object proposal method results in more running time. We consider exploring how to optimize the object proposal method for reducing running time and apply our aircraft detection framework to other target detection tasks in future work.

Acknowledgments: This work is supported partly by the National Natural Science Foundation of China under Grant 41501485. The authors would like to thank the three reviewers and academic editor for their constructive comments and helpful suggestions.

Author Contributions: Yang Li, Hao Sun and Kun Fu conceived and designed the experiments; Yang Li and Hao Sun performed the experiments; Yang Li, Hao Sun and Xian Sun analyzed the experiments; Yang Li wrote the paper.

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

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
2. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1440–1448.
3. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
4. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Amsterdam, The Netherlands, 2016; pp. 21–37.
5. Wu, H.; Zhang, H.; Zhang, J.; Xu, F. Fast aircraft detection in satellite images based on convolutional neural networks. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 27–30 September 2015; pp. 4210–4214.
6. Zhang, Y.; Fu, K.; Sun, H.; Sun, X.; Zheng, X.; Wang, H. A multi-model ensemble method based on convolutional neural networks for aircraft detection in large remote sensing images. *Remote Sens. Lett.* **2018**, *9*, 11–20.
7. Caicedo, J.C.; Lazebnik, S. Active object localization with deep reinforcement learning. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 2488–2496.
8. Bellver, M.; Giró i Nieto, X.; Marqués, F.; Torres, J. Hierarchical Object Detection with Deep Reinforcement Learning. *arXiv* **2016**, arXiv:1611.03718. Available online: http://arxiv.org/abs/1611.03718 (accessed on 1 February 2018).
9. Jie, Z.; Liang, X.; Feng, J.; Jin, X.; Lu, W.; Yan, S. Tree-structured reinforcement learning for sequential object localization. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 127–135.
10. Sun, H.; Sun, X.; Wang, H.; Li, Y.; Li, X. Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model. *IEEE Geosci. Remote Sens. Lett.* **2012**, *9*, 109–113.
11. Zhang, W.; Sun, X.; Fu, K.; Wang, C.; Wang, H. Object detection in high-resolution remote sensing images using rotation invariant parts based model. *IEEE Geosci. Remote Sens. Lett.* **2014**, *11*, 74–78.
12. Liu, G.; Sun, X.; Fu, K.; Wang, H. Aircraft recognition in high-resolution satellite images using coarse-to-fine shape prior. *IEEE Geosci. Remote Sens. Lett.* **2013**, *10*, 573–577.
13. Diao, W.; Sun, X.; Zheng, X.; Dou, F.; Wang, H.; Fu, K. Efficient saliency-based object detection in remote sensing images using deep belief networks. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 137–141.
14. Chen, X.; Xiang, S.; Liu, C.L.; Pan, C.H. Aircraft Detection by Deep Convolutional Neural Networks. *IPSJ Trans. Comput. Vis. Appl.* **2015**, *7*, 10–17.
15. Xu, T.B.; Cheng, G.L.; Yang, J.; Liu, C.L. Fast Aircraft Detection Using End-to-End Fully Convolutional Network. In Proceedings of the IEEE International Conference on Digital Signal Processing, Beijing, China, 16–18 October 2016; pp. 139–143.
16. Wang, S.; Gao, X.; Sun, H.; Zheng, X.; Xian, S. An Aircraft Detection Method Based on Convolutional Neural Networks in High-Resolution SAR Images. *J. Radars* **2017**, *6*, 195.
17. Xu, Z.; Wang, R.; Li, N.; Zhang, H.; Zhang, L. A Novel Approach to Change Detection in SAR Images with CNN Classification. *J. Radars* **2017**, *6*, 483.
18. Zhao, J.; Guo, W.; Liu, B.; Cui, S.; Zhang, Z.; Yu, W. Convolutional Neural Network-based SAR Image Classification with Noisy Labels. *J. Radars* **2017**, *6*, 514.

19. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M.A. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602. Available online: http://arxiv.org/abs/1312.5602 (accessed on 1 February 2018).
20. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–489.
21. Abbeel, P.; Coates, A.; Ng, A.Y. Autonomous helicopter aerobatics through apprenticeship learning. *Int. J. Robot. Res.* **2010**, *29*, 1608–1639.
22. Van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-Learning. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), Phoenix, AZ, USA, 12–17 February 2016; pp. 2094–2100.
23. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. In Proceedings of the 31st International Conference on Machine Learning (ICML-14), Beijing, China, 21–26 June 2014; pp. 387–395.
24. Yun, S.; Choi, J.; Yoo, Y.; Yun, K.; Choi, J.Y. Action-Decision Networks for Visual Tracking with Deep Reinforcement Learning. In Proceedings of the 2017 Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1349–1358.
25. Jayaraman, D.; Grauman, K. Look-ahead before you leap: End-to-end active recognition by forecasting the effect of motion. In *European Conference on Computer Vision*; Springer: Amsterdam, The Netherlands, 2016; pp. 489–505.
26. Hosang, J.; Benenson, R.; Dollár, P.; Schiele, B. What makes for effective detection proposals? *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 814–830.
27. Cheng, M.M.; Zhang, Z.; Lin, W.Y.; Torr, P. BING: Binarized normed gradients for objectness estimation at 300fps. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 3286–3293.
28. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171.
29. Zitnick, C.L.; Dollár, P. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*; Springer: Zurich, Switzerland, 2014; pp. 391–405.
30. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533.
31. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861. Available online: http://arxiv.org/abs/1704.04861 (accessed on 1 February 2018).
32. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556. Available online: http://arxiv.org/abs/1409.1556 (accessed on 1 February 2018).
33. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. Available online: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/ (accessed on 1 February 2018).
34. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. Available online: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/ (accessed on 1 February 2018).