remote sensing

MDPI

Article

Unified Partial Configuration Model Framework for Fast Partially Occluded Object Detection in High-Resolution Remote Sensing Images

Shaohua Qiu [1,*], Gongjian Wen [1], Jia Liu [2,*], Zhipeng Deng [3] and Yaxiang Fan [1]

1. Science and Technology on Automatic Target Recognition Laboratory, National University of Defense Technology, Changsha 410073, China; wengongjian@sina.com (G.W.); fanyaxiang@126.com (Y.F.)
2. College of Meteorology and Oceanology, National University of Defense Technology, Changsha 410073, China
3. College of Electronic Science, National University of Defense Technology, Changsha 410073, China; zpdeng@whu.edu.cn

* Correspondence: qiushaohua11@nudt.edu.cn (S.Q.); liujia2016@nudt.edu.cn (J.L.); Tel.: +86-731-8457-6246 (J.L.)

Received: 15 December 2017; Accepted: 13 March 2018; Published: 15 March 2018

Abstract: Partially occluded object detection (POOD) has been an important task for both civil and military applications that use high-resolution remote sensing images (HR-RSIs). This topic is very challenging due to the limited object evidence for detection. Recent partial configuration model (PCM) based methods deal with occlusion yet suffer from the problems of massive manual annotation, separate parameter learning, and low training and detection efficiency. To tackle this, a unified PCM framework (UniPCM) is proposed in this paper. The proposed UniPCM adopts a part sharing mechanism which directly shares the root and part filters of a deformable part-based model (DPM) among different partial configurations. It largely reduces the convolution overhead during both training and detection. In UniPCM, a novel DPM deformation deviation method is proposed for spatial interrelationship estimation of PCM, and a unified weights learning method is presented to simultaneously obtain the weights of elements within each partial configuration and the weights between partial configurations. Experiments on three HR-RSI datasets show that the proposed UniPCM method achieves a much higher training and detection efficiency for POOD compared with state-of-the-art PCM-based methods, while maintaining a comparable detection accuracy. UniPCM obtains a training speedup of maximal 10× and 2.5× for airplane and ship, and a detection speedup of maximal 7.2×, 4.1× and 2.5× on three test sets, respectively.

Keywords: high-resolution remote sensing images; partially occluded object detection; partial configuration model; unified detection framework; part sharing; deformable part-based model

1. Introduction

Object detection has been a key factor in high-resolution remote sensing images (HR-RSIs) analysis, and has been extensively studied for the remote sensing community. It is basic but challenging, due to the fact that HR-RSIs contain objects with different textures, shapes, and complex backgrounds and are always affected by different imaging conditions caused by weather, clouds, sun elevation angles, etc.

Due to recent advancement in remote sensing technology, the detection of small moveable manmade objects becomes possible, and the focus of object detection in remote sensing has been gradually moved to relatively small targets, such as vehicles, airplanes, and ships, from large man-made infrastructures such as airports and residual areas. This shift leads to a situation in which, on the one hand, the induced information in HR-RSIs can easily affect the detection accuracy by the background, and on the other hand makes it hard to find frequently partially occluded objects with missing information. This paper exactly concentrates on partially occluded object detection (POOD) in HR-RSIs with high accuracy.

The state-of-the-art object detection methods for HR-RSIs can be roughly categorized into two groups. The first group of methods are based on deep learning (mainly the convolutional neural network (CNN) based methods [1–10]), which use a large amount of samples to train a deep layered neural network and learn deep feature representation to classify the candidate regions into objects and background. Its performances on ordinary unoccluded HR-RSI datasets is remarkable and superior. However, state-of-the-art CNN-based methods are unable to precisely handle the occlusion problem. Although some tricks [11] can be applied to tackle it to some extent, its capability is rather limited. For example, the data augmentation method [12] is always used to extend the training set to account for different transformations of objects, such as sample cropping. However, the trained network can only detect the unoccluded object area and will be unable to infer the exact location of the partially occluded object, which is considered to be the essential quality for an object detection method. Moreover, its black-box architecture also makes it relatively hard to accommodate heavy occlusion [13] which is very common in HR-RSIs.

The other group of methods have handcrafted features or handcrafted architectures [13–22]. They either concentrate on better feature representation for objects [18,19], or focus on flexibly modeling the object structures and appearance [17,20], or try to design a better classifier [21,22]. This group has been extensively studied in previous decades before the emergence of deep learning-based methods. The deformable part-based method (DPM) proposed by Felzenszwalb [15] is an outstanding generic object detection method in this group. It models object structure with a pictorial structure model and object appearance with part and root filters. Successively, researchers introduce DPM into object detection in HR-RSIs [14,16,20,23,24]. However, researchers demonstrate that the performance of DPM will drop greatly when occlusion happens [17,25]. The fact is that the detection tasks in varied applications using HR-RSIs are always accompanied with large occlusion.

To tackle this, Ouyang et.al. and Niknejad et al. [26,27] both firstly utilize DPM to detect the scores of parts and then infer the occlusion with a hierarchical deep belief network and a two-layer conditional random field. These two methods concentrate on the inference of occlusion based on the scores of parts at hand. Different to these, our previous work [17] proposed a two-layer object model named partial configuration object model (PCM) to deal with the occlusion problem in HR-RSIs. It designs several partial configurations to detect the unoccluded evidences and finally synthesizes these evidences to justify the presence of an object. Its idea is very simple and the result is promising. However, it requires a great amount of manual predefinition and massive computation [13]. To solve the problem of PCM, an automatic and fast PCM generation method, AFI-PCM, was proposed by us in [13]. It proposed a part sharing mechanism to tackle the above problem, thereby making the automatic implementation of PCM possible. However, it only pays attention to the model training stage of PCM and the more concerned detection stage is exactly as slow as PCM.

The work in this paper is inspired from the problems in PCM and AFI-PCM, which tries to integrate the entire PCM model with a novel part sharing mechanism for both fast and automatic model training and detection, that is, a unified PCM framework for fast POOD. To the best of our knowledge, no PCM/AFI-PCM–like POOD methods are reported currently, except for Refs. [13,17]. The work in this paper is a largely extended version of our previous work [13,17]. The contributions of the work can be found in four aspects: (1) we analyze the shortcomings of PCM and AFI-PCM in detail, and give the inherent causes that lead to these shortcomings; (2) we propose to use a part sharing mechanism for fast POOD, which will get around the PCM assembly process. It makes the entire training and detection processes a unified framework, and obtains great detection speedup compared to PCM and AFI-PCM. Experimental results not only verified the speedup, but also show that it can obtain a comparable accuracy with PCM; (3) we propose a novel spatial interrelationship estimation method directly from the deformation information of DPM model; (4) we propose to simultaneously learn the weights within a partial configuration and the weights between partial configurations in a unified way.

The remainder of this paper is organized as follows. Section 2 analyzes the shortcomings of PCM and AFI-PCM in detail and gives their inherent causes. Section 3 gives detailed information about the proposed unified PCM framework for fast POOD. Section 4 shows the experimental results. And conclusions are drawn in Section 5.

2. Shortcomings Analysis

In this section, we first briefly introduce PCM and AFI-PCM which are both designed for POOD in HR-RSIs, and then give detailed shortcoming analysis of these two methods. For convenience, we take the airplane category as an example to elaborate all related ideas hereinafter.

2.1. Brief Review of PCM and AFI-PCM

Compared to conventional single-layer object detection methods like DPM, PCM defined an extra buffer layer to block the occlusion impact from passing onto the object layer. This layer is intended to encode possible occlusion states. It is named as partial configurations which is the configuration of adjacent predefined semantic parts. Each partial configuration is represented by a standard DPM model.

The basic unit in PCM is semantic part. They are selected from a predefined category-dependent undirected skeleton graph. The definition of semantic parts determines the coverage of partial configurations. The coverage will then be used to estimate the spatial interrelationship that converts partial configuration hypotheses into full-object ones in the second layer of PCM. The second layer is the object layer that is the arrangements of partial configurations. Instead of the commonly used linear-scoring mechanism, a max-scoring one is used in the object layer to ensure that even certain partial configurations are occluded; the remained partial configurations will still have the chance to capture the unoccluded evidence of the presence of object to finally detect the object. During detection, unoccluded partial configurations will contribute to the locating of the partially occluded objects.

AFI-PCM mainly aims at the training stage for automatic and fast implementation of PCM. Based on the idea that a trained DPM model contains all the elements that PCM needs, AFI-PCM proposes to share part and root filters within partial configurations. It means that AFI-PCM find another way around to directly assemble these elements to generate new PCM models, while still following the detection pipeline of PCM. AFI-PCM adds an extra DPM filter layer into the two-layer object architecture of PCM for training, and uses the same two-layer architecture for model detection.

2.2. Shortcomings of PCM

In this section, we extend the analyses in [13] about shortcomings of PCM from four perspectives.

2.2.1. Uncertainty of Semantic Parts and the Gap between Semantic Parts and Parts in Detection

The semantic part is the basic element in the first layer, of which the combination determines the coverage of partial configurations. The semantic parts are defined manually and semantically, along with the skeleton graph of the category. Generally, it is hard to determine whether the defined semantic parts are suitable for detection.

Actually, the semantic parts are not explicitly used in the partial configuration layer. They are only used to calculate the coverage of partial configuration. Partial configuration is represented by a standard DPM with parts located at salient points of the object. Therefore, these semantic parts are not the parts that we use for detection. In this way, the human prior does not contribute to the detection. Figure 1 shows all related parts in the entire train and detection chain. It can be seen that semantic parts are different from the parts trained in the DPM model. It also indicates that the entire PCM model is designed with redundancy, as large amount of extra work will be needed to transform the defined semantic parts to the trained parts.

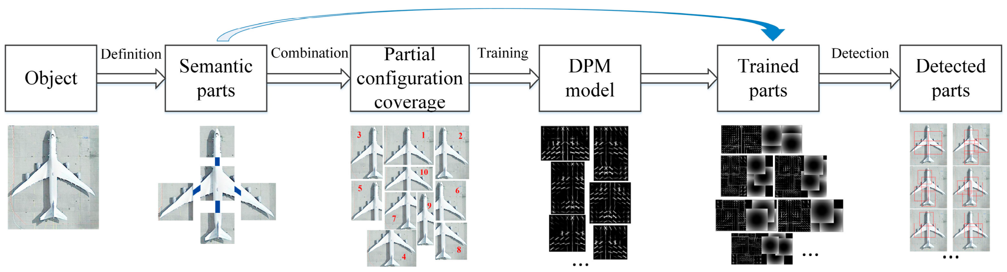


Figure 1. Relevant parts in partial configuration model (PCM). Semantic parts are designed from the object, while the trained parts are from the deformable part-based model (DPM) trained from the samples of partial configurations whose coverage is defined by semantic parts. The detection of trained parts on images will generate detected parts.

2.2.2. Tedious Sample Annotation and Arbitrary Skeleton Graph Predefinition

As revealed above, obtaining the semantic parts becomes a necessary and unavoidable job in PCM. Moreover, we cannot use definite rules to obtain partial configurations' coverage directly from the bounding box of a full object, in order to avoid the variances of Gaussian functions of a spatial interrelationship being zero and the following weighted continuous clustering being discretized.

For PCM, annotating the semantic parts of each training sample becomes an inevitable route. However, this work is highly labor-intensive. Though it tries to reduce the annotation volume of a full object through semantic part combination, a large amount of additional annotations is still needed, compared to general object detection methods which only need full-object annotations.

In PCM, the semantic parts are organized with a category-dependent skeleton graph, and this graph should be defined beforehand for each category by operators. This predefinition relies too much on the operators' subjective understanding, and sometimes it is also hard to tell whether the definition is suitable for detection. The predefinition of the skeleton graph greatly hinders the automatic implementation of PCM to other new object categories.

2.2.3. Repeated Training of Shared Semantic Part Areas

PCM uses multiple partial configurations to handle the occlusion impact. All these partial configurations are separately trained using the samples cropped from the same full object samples according to the coverage combined from the same set of semantic parts. This multi-model training process is actually a repeated process of modeling the same set of semantic part areas, as can be observed in Figure 1. Therefore, the training of PCM is inefficient.

2.2.4. Repeated Detection of Shared Semantic Part Areas

During the detection stage, each DPM model of partial configuration will slide through the entire image to obtain the corresponding responses of part and root filters. Similarly, for an object to be detected, the multi-model detection process will convolve through its shared semantic part areas 10 times if the number of partial configurations is 10. Therefore, the detection process of PCM is also rather time-consuming. As can be imagined, it will be unsuitable for the applications that online detection speed is crucial for, especially when using very large HR-RSIs.

2.3. Shortcomings of AFI-PCM

2.3.1. Repeated Part Detection and Redundant PCM Packing

AFI-PCM in [13] tries to speed up the training stage of PCM through a part sharing mechanism. It intends to share the same set of parts from a trained DPM model of a full object, and pack the shared part filters into new PCM models. As a consequence, its detection process is exactly the same as with PCM, and it shares the fourth shortcoming of PCM, that is, the more important detection stage is ignored.

We take a detection example of a partial configuration and Figure 2 illustrates the detection time consumption percentages of the four main steps: construct feature pyramid, obtain filter response, apply structural rules, and get final detections. The first two steps take more than 75% of the detection computation, while this work is exactly the same for different PCM models for the same image. The remaining two steps are different between different partial configurations but only take less than a quarter of the time. If the first two steps are shared, the computation overhead will be greatly reduced.

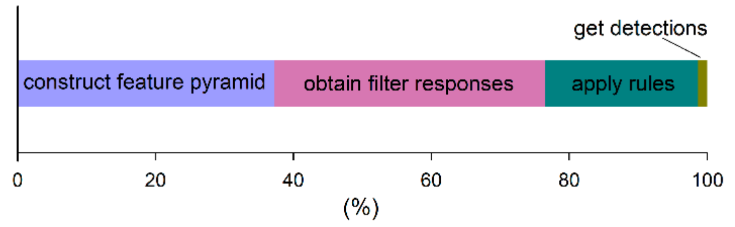


Figure 2. Time consumption of different steps in the detection of a partial configuration. As each partial configuration is represented by a DPM model, its detection is exactly a standard DPM detection process which is to (1) construct feature pyramid; (2) obtain filter responses; (3) apply rules, and (4) get detections. The first two steps take up nearly three quarters of the time consumption.

2.3.2. Separated Weights Learning among and within Partial Configurations

In AFI-PCM, there are two key factors. One is the weights that balance the performance differences between partial configurations, and the other one is the weights that balance the elements within a partial configuration. In PCM, the former is obtained through a max-margin Support Vector Machine (SVM) optimization method and the latter is done within the DPM method through latent SVM. In AFI-PCM, both parameters are explicitly optimized by the same max-margin SVM optimization method. However, these two optimization processes are separately learnt on two different datasets. Therefore, the optimization error introduced by the first round of optimization will be passed to the second round of optimization and it will be magnified.

2.3.3. Extra Loop in Spatial Interrelationship Estimation

In PCM, the spatial interrelationship is estimated from offsets statistics from the manually annotated partial configuration to the full object. This annotation process is tedious, as analyzed in the above section. In AFI-PCM, this process is made into an automatic one. It uses the trained DPM model to re-detect the training samples to obtain the part locations for spatial interrelationship estimation. However, there is an extra loop in this process (See the blue arrows in Figure 3), although already better than manual annotation. DPM learns the part deformation from massive training samples, and then it is used to redetect the training samples for part location variations. These variations are finally used to estimate the spatial interrelationship. The evident loop also takes a great amount of time in the training stage.

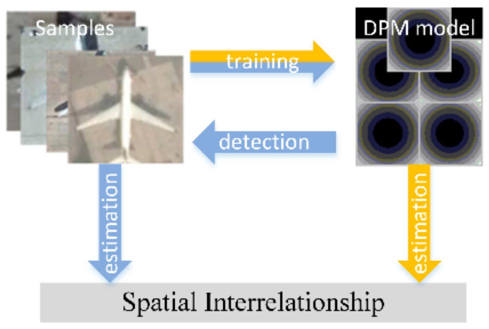


Figure 3. Spatial interrelationship estimation loop in AFI-PCM.

2.4. What Are the Causes of These Shortcomings in PCM and AFI-PCM?

PCM consists of two layers. The first layer is the partial configuration layer, and the second layer is the object layer. There is also a hidden layer in PCM, the semantic part layer, which is the basis of the first layer. The architecture of PCM is given in Figure 4a. The number of subjects/items in the first layer is much greater than other layers. As revealed above, the main body of all training or detection work is involved in the partial configuration layer, and the semantic part layer is only for the purpose of data preparation. This leads to a situation where a large number of subjects need to be trained or detected for the first layer. Although the elements in the partial configuration layer are based on the same basis layer, they do not share the basis during the training and detection, that is, the entire basis layer is separately trained or detected for each partial configuration. We believe that this is where these shortcomings in PCM come from.

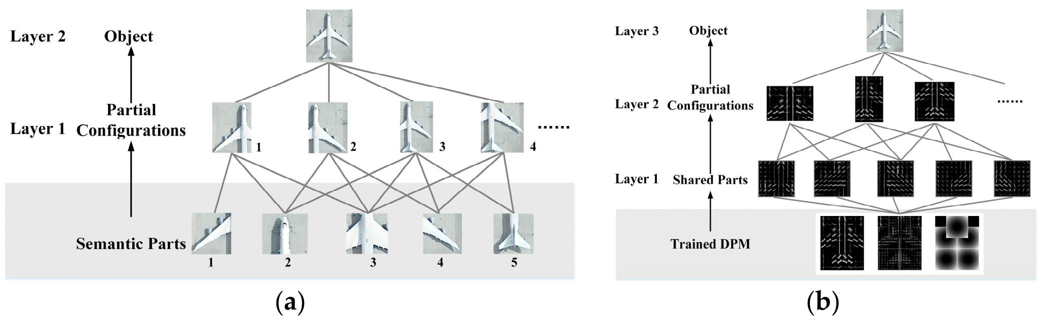


Figure 4. The architectures of (**a**) PCM and (**b**) unified PCM for fast partially occluded object detection (POOD).

As for AFI-PCM, it has a different architecture from PCM. It uses the part layer as its basis layer, and all other layers are constructed from this. However, this benefit is only used for the training stage, not for the more important detection stage. A more unified framework for both training and detection is preferred, as well as the unified parameter learning of different layers. Another cause is that the potential of the trained DPM is not fully explored. For example, the deformation information of an object is already trained from massive samples, so it does not need to re-detect training samples for this information.

3. Unified PCM Framework for Fast POOD

Given the analysis of the shortcomings of PCM and AFI-PCM and their causes, we propose to "sink" the main body of all training and detection work into its lower layer, as indicated in Figure 4b. We add an extra shared part layer under the original partial configuration layer. The trained DPM model becomes the hidden layer in our architecture. In PCM, all data preparation work, such as annotations, is included in the hidden layer, and the main body of training and detection lies in the first layer. In the proposed architecture, the intention is similar: the data preparation work is for the hidden layer of training DPM and the main body of training and detection goes to the first layer for shared parts training and detection. We also keep the merits of PCM, which uses fully visible samples to train PCM for partially occluded objects.

In this section, the proposed unified PCM framework (UniPCM) for fast POOD is described in detail. The proposed framework is illustrated in Figure 5, which can be divided into several aspects. For the training stage, the original multi-model training process is replaced by (1) single model training; (2) partial configuration design; (3) spatial interrelationship estimation; and (4) unified weights learning. The multi-model detection stage is changed into a single model detection and part sharing based results combination. We firstly give the motivation for a part sharing mechanism, and then describe the key points of the proposed UniPCM framework for fast POOD.

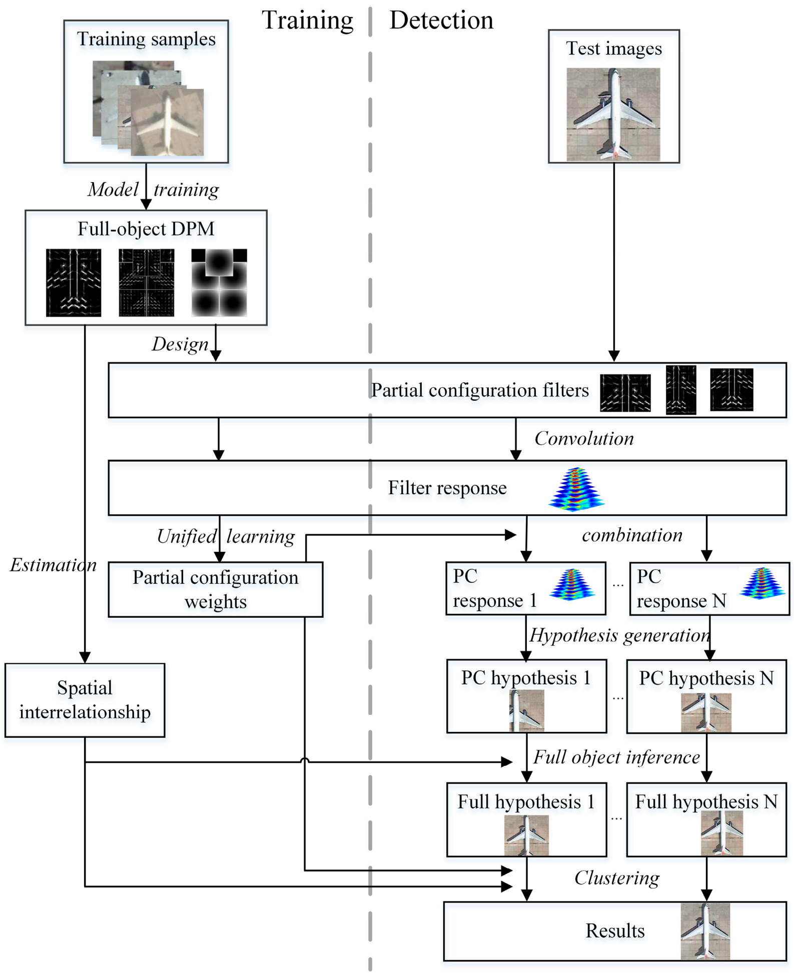


Figure 5. The UniPCM framework for fast POOD. The framework can be divided into two stages: training and detection. The training stage includes full-object DPM training, partial configuration design, and unified weights learning, and spatial interrelationship estimation. The detection stage consists of shared filter convolution, response combination, hypothesis generation, full object inference, and clustering.

3.1. Part Sharing Mechanism

We first take a look at the first shortcoming of PCM. Once the coverage of partial configuration is set, all related parts are consistent in the following training and detection processes, that is, no gap exists anymore, as can be seen in Figure 1. Since the parts of DPM cannot be annotated beforehand, PCM uses the semantic parts with similar functions to replace them. This is the cause of the second shortcoming. If we can use the same set of parts for both annotation and training, the gap between the semantic parts and the parts in DPM could be bridged.

Once we obtain a DPM model, the coverage of partial configurations can be determined based on its parts' locations. Hence, in this paper, we propose to directly use the parts trained from DPM as our semantic parts. Actually, this DPM model is a full-object model trained from fully visible samples. Although the parts of a full object DPM model have no semantic meaning, they do cover certain salient areas of the object. Their combination can also represent occlusion patterns which are most related to POOD using this DPM model. In this way, no skeleton graph has to be defined beforehand since the spatial configuration of parts in DPM is already trained from samples.

For the third shortcoming, the overlapping areas of partial configurations have already been trained, so there is no need to re-train them if we directly use the parts from the trained DPM model.

Accordingly, we propose to directly share these part filters for partial configurations. In this way, only one model needs to be trained, which will largely speed up the training process.

The fourth shortcoming of PCM is similar to the third one. In PCM, each partial configuration will detect their own parts. Once the parts are shared, only one set of original parts needs to be detected, and the results of partial configurations can thus be inferred from them, which is the same situation with AFI-PCM.

As revealed above, the part sharing mechanism mainly operates for/at the parts of PCM, and successfully "sinks" the main body of training from the partial configuration layer in PCM (Figure 4a) to the shared part layer in Figure 4b. Consequently, part sharing exactly solves the shortcomings mentioned above for PCM and AFI-PCM, and will largely speed up training and detection simultaneously.

3.2. Partial Configuration Design

Different from PCM that uses predefined semantic parts and a skeleton graph to generate partial configurations, all that is available for our method is the trained full-object DPM model. As each partial configuration is represented by a DPM model, we need to prepare the root filter, part filters, and deformation information for a new partial configuration model, which is described in the following subsections.

3.2.1. Part Selection

The first thing is to determine which parts to be combined into the new model. Each part in DPM model is characterized by its deformation anchor and size, which is a good guide for our determination. Without the skeleton graph, there are arbitrary possibilities of part combination. There are always many unreasonable combinations, representing the occlusion patterns that are not likely to happen in applications. For the case of remote sensing, geospatial objects on the ground are more likely to be occluded by larger occluders, such as clouds or its functional facilities. The occlusion patterns, such as a hollowed-out occlusion pattern, are unlikely to happen or very rare. In PCM, these unreasonable patterns are filtered out with the aid of a predefined skeleton graph. In our work, the only guide is the parts' locations. Therefore, we relax the skeleton graph to a more generalized one. We form a net-like graph based on the anchor locations of parts, and then choose all possible sub-graphs from this graph, like PCM under connectivity constraints to leave out unreasonable possible combinations.

For a trained DPM model with n parts represented by $\boldsymbol{P}_j = \left\{\boldsymbol{p}_j, \boldsymbol{l}_j, \boldsymbol{w}_j\right\}$, $j = 1, \cdots, n$ where $\boldsymbol{p}_j$ corresponds to the part filter parameters, $\boldsymbol{l}_j = \left(l_{j,x}, l_{j,y}\right)$ is its anchor position relative to the root and $\boldsymbol{w}_j$ is the deformation parameter; the location set V can be formed into a new undirected and connected graph $G(V, E)$, see Equation (1) and Figure 6. It is implemented by the iterative Delaunay triangulation method in [28].

$$\begin{cases} V = \left\{\boldsymbol{l}_j \middle| j = 1, \cdots, n\right\} \\ E \subseteq \{(x, y) | x, y \in V, x \neq y\} \end{cases} \tag{1}$$

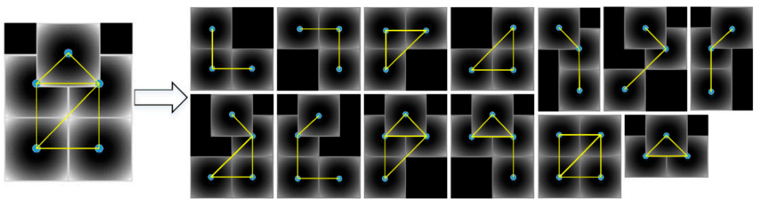


Figure 6. Delaunary triangulation of parts' locations and corresponding sub-graph selection.

The main difference between this net-like graph and the skeleton graph in PCM is that the former is generated automatically according to parts' locations and the latter is defined by human operator. This graph is more consistent with the DPM model for detection, as analyzed in Section 2.

When the graph is settled down, the partial configuration design process is the same with [13,17] which is to select a sub-graph from $G(V, E)$ while maintaining the connectivity constraint [29] and the vertex number constraint. These two constraints are used to model the location and ratio of occlusion. An example of this graph is given in Figure 6. This process can be formulated as automatically choosing a connected sub-graph $G(V_A, E_A)$ from $G(V, E)$ as Equation (2) and the selected part index set is formulated as A.

$$A = \{i|i = 1, \cdots, n\}, \text{ where } \begin{cases} |2 < \text{card}(A) < n \\ |G(V_A, E_A) \subset G(V, E) \\ |G(V_A, E_A) \text{ is connected} \\ |E_A \subseteq \{(x, y)|x, y \in V_A, x \neq y\} \subset E \end{cases} \tag{2}$$

When the part index set of each partial configuration is determined, we directly select the corresponding part parameters of $\{\boldsymbol{p}_i\}(i \in A)$ from the trained DPM model as the parts of the partial configuration and keep their deformation parameters unchanged.

3.2.2. Root Cropping

The next step is to generate the root filter of partial configuration. The root filter for DPM is a parameter matrix that will convolve through the feature space to obtain the global appearance response of the object. For a partial configuration, its coverage is calculated from the minimal bounding rectangle of its selected parts. Then we directly crop the root filter parameter matrix according to this calculated coverage. The obtained root filter parameter matrix will be reweighted in the following step. Once the new root filter is set, the deformation anchors of parts are shifted to their actual locations.

Up until now, we have completed the work for selecting parts and cropping root from the trained DPM model. The last step away from a complete DPM model is the balance between the selected parts and cropped root.

3.3. Unified Weights Learning

In PCM, partial configurations are unbalanced and need to be weighted. We call this process inter-partial-configuration (inter-PC) weights learning which is conducted on the validation samples using a max-margin SVM framework. Actually, the shared filters in a partial configuration are also unbalanced and need to be reweighed as well. We call this intra-partial-configuration (intra-PC) weights learning. To combine these two processes into the entire framework of PCM, a natural idea is to coarsely aggregate these two reweighing processes. However, intra-PC parameters are intermediate results, and will be implicitly used in inter-PC weights learning. Moreover, they are two separated processes conducted on two different sample sets. The quality of intra-PC weights learning will largely affect the followed inter-PC weights learning.

We first review the weights learning process of PCM as illustrated in Figure 7. In each partial configuration, each component will be assigned a weight $w_{i,j}$, including a bias term b_i. When the N partial configurations have settled down, each partial configuration will be assigned a weight T_i to indicate its importance during the following hypotheses fusion process. In inter-PC weights learning, these scores of partial configurations will be combined with a linear formula. One simple idea is to directly optimize these two processes in one step. However, when the weights are multiplied for inter-PC optimization, it becomes a non-convex problem which is hard to optimize.

To tackle this problem, we propose a unified weights learning framework in this paper, which takes another way around. To separate the multiplied weights for a convex problem, we change the second optimization into a filter reweighing problem, that is, optimizing the weights of all cropped root filters and all part filters, and then obtaining the weights of partial configurations from these learnt filter weights.

For each partial configuration, we optimize the weights of its elements. Let s_j denotes the score of filter $j (j \in A_i)$ of partial configuration $i (i = 1, \cdots, N)$, the score of the partial configuration can be formulated as

$$y_i = \sum_{j \in A_i} w_{i,j} s_j + b_i = \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{s}_{A_i} + b_i \tag{3}$$

The optimization objective function is that the weights make the combination way in Equation (3) match their labels on validation samples. For all prepared filters from the trained DPM model, their score combination should also match their labels l. Let $s_k (1 \le k \le n + N)$ denote the score of k-th filter (including part filters and cropped root filters), their final label can be formulated as

$$Y = \sum_{k=1}^{n+N} \alpha_k s_k + B = \boldsymbol{\alpha}^T \boldsymbol{s} + B \tag{4}$$

where α_k is the weights of two kinds of filters, and B is the bias term. We combine the above two optimization processes into one optimization process that simultaneously obtains the intra-PC and inter-PC weights which are learnt by minimizing the structured prediction objective function:

$$\min_{\boldsymbol{w}_i, \boldsymbol{\alpha}, b, B} \frac{1}{2} \left(\boldsymbol{\alpha}^T \boldsymbol{\alpha} + \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{w}_i^T \boldsymbol{w}_i \right) + \sum_{i=1}^{N} C_i \sum_j \epsilon_j + C \sum_j \xi_j, \text{ s.t. } \begin{cases} \forall j,\ y_{i,j} l_j \ge 1 - \epsilon_j \\ \forall j, Y_j l_j \ge 1 - \xi_j \\ \boldsymbol{w}_i \ge 0,\ \epsilon_j \ge 0 \\ \boldsymbol{\alpha} \ge 0,\ \xi_j \ge 0 \end{cases} \tag{5}$$

This standard convex optimization problem can be solved by Lagrange multipliers method using the standard off-the-shelf convex optimization package CVX [30]. The constants C_i, C can be found using 5-fold cross validation.

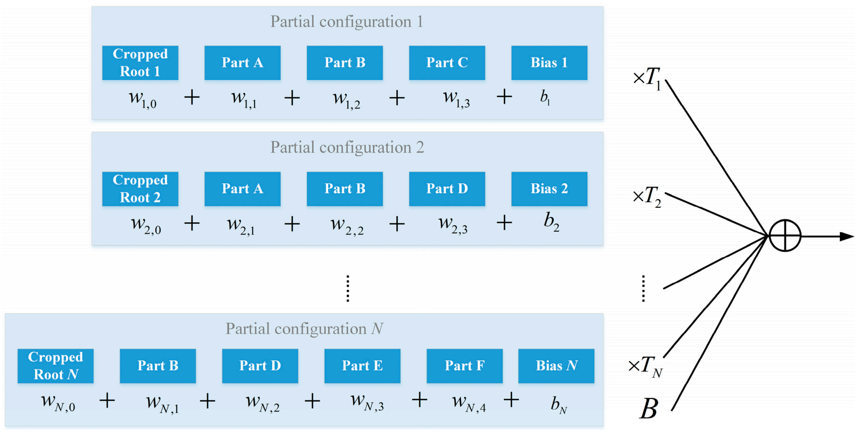


Figure 7. Weights learning of PCM which includes intra-PC weights learning and inter-PC weights learning.

The relationship between these weights can be formulated as

$$\begin{cases} T_i w_{i,1} = \alpha_i,\ 1 \le i \le N \\ \sum_z^N \mathbb{I}(j \in A_z) T_z w_{z,j+1} = \alpha_{j+N},\ 1 \le j \le n \\ \sum_k^N T_k b_k + T_B = B \end{cases} \tag{6}$$

where $\mathbb{I}(true) = 1, \mathbb{I}(false) = 0$. Once we obtain the above weights $\boldsymbol{w}_i$, $\boldsymbol{\alpha}$, the inter-PC weights T_i can be calculated from a least-square solution.

3.4. Spatial Interrelationship Estimation

Through samples re-detection, the partial configuration location distribution can be estimated from the relative offsets of detected parts in AFI-PCM, which is of low efficiency as analyzed in Section 2. These offset variances actually come from the deformation ability embedded in the DPM model as a match cost like a spring. These deformation parameters are trained from massive training samples, and it is a repetitive process to re-detect the deformation related variances. Based on this fact, we propose to derive the spatial interrelationship directly from the deformation parameters trained in DPM for partial configurations (indicated as yellow arrows in Figure 3). Its main idea is described as follows.

Given the index set A of a partial configuration, we use their corresponding parts $\boldsymbol{P}_j = \left\{\boldsymbol{p}_j, \boldsymbol{l}_j, \boldsymbol{w}_j\right\}$, $j \in \mathrm{A}$ to derive the location interrelationship and size interrelationship. The deformation cost in DPM for location $\boldsymbol{x} = (x, y)$ is defined as

$$d = \Delta\boldsymbol{D} \cdot \boldsymbol{w}_j, \text{ where } \begin{cases} \Delta\boldsymbol{D} = \left(\Delta x, \Delta y, \Delta x^2, \Delta y^2\right) \\ (\Delta x, \Delta y) = \boldsymbol{l}_j - \boldsymbol{x} = \left(l_{j,x}, l_{j,y}\right) - (x, y) \end{cases}. \tag{7}$$

This deformation cost performs like a spring, and the closer the part locates to the ideal anchor position, the lower the cost will be. This inspires us to transform the deformation cost into a location distribution term. We use a simple conversion function in Equation (8) where $\boldsymbol{l}$ is the anchor position, $\boldsymbol{x}$ is the current position, and $\boldsymbol{w}$ is the deformation parameter vector. $N(\cdot)$ is a normalization function that constrains the data in the range of part size and convert the value range to [0, 1]. Based on the above function, the part location becomes a continuous spatial probability density function (PDF). As can be seen in Equation (7), the deformation cost in DPM is dimension-independent, so the following derivation process can be conducted independently in X and Y dimensions.

$$f_{\boldsymbol{l},\boldsymbol{w}}(x) = -N((\boldsymbol{l} - \boldsymbol{x}) * \boldsymbol{w}) \tag{8}$$

The location interrelationship represents the relative shift from the center of partial configuration to the object center, and the size interrelationship represents the size ratio between partial configuration and the full object. Given the index set A of a partial configuration and its $\boldsymbol{P}_j, j \in A$, the interrelationships can be estimated as

$$\begin{cases} |\tau_x = \frac{(R+L)-(r+l)}{2(r-l)} \\ |\tau_y = \frac{(B+T)-(b+t)}{2(b-t)} \\ |\tau_w = \frac{R-L}{r-l} \\ |\tau_h = \frac{B-T}{b-t} \end{cases}, \text{ where } \begin{cases} |R = max\left\{\boldsymbol{l}_{j,x}\right\} \\ |L = min\left\{\boldsymbol{l}_{j,x}\right\} \\ |B = max\left\{\boldsymbol{l}_{j,y}\right\} \\ |T = min\left\{\boldsymbol{l}_{j,y}\right\} \end{cases} \text{ and } \begin{cases} |r = max\{\boldsymbol{l}_{m,x}\} \\ |l = min\{\boldsymbol{l}_{m,x}\} \\ |b = max\{\boldsymbol{l}_{m,y}\} \\ |t = min\{\boldsymbol{l}_{m,y}\} \end{cases}, m \in A,\ j = 1, \cdots, n \tag{9}$$

In Equation (9), (R, L, B, T) and (r, l, b, t) represent the right, left, bottom, and top of the bounding boxes of full object and partial configuration. In this way, the continuous interrelationship estimation can be solved as the derivation from the PDFs of parts to the PDF of $\left(\tau_x, \tau_y, \tau_w, \tau_h\right)$ based on Equation (9). Fortunately, most of the operations are linear, and they can be directly solved with cascaded operations of PDF estimations. All related formulations of the operations are given in Appendix A.

3.5. Unified Detection Framework

In PCM, the obtained models of partial configurations individually slide through the entire image to obtain their corresponding detection results, which means that the computational cost is nearly N times as much as that of a full-object DPM model. This paper aims to cut down the computation in both the training and detection stage. Along with the training scheme, our detection is also based on the part sharing mechanism in a unified detection framework, which will largely reduce the computational overhead.

Similar to the training of PCM in that only one DPM model needs to be trained, our detection also needs only one DPM detection with similar computational cost, thereby largely reducing the cost, compared to PCM and AFI-PCM. It means that only one DPM will slide through the image pyramid, and all the POOD work is done based on the results of this DPM.

For a trained DPM with n parts, we can obtain N partial configurations. Therefore, N cropped root filters and n part filters are prepared for detection from the trained DPM model, along with their weights. In the proposed method, we only have to get the responses of $(N + n)$ filters and then all responses of partial configurations are obtained.

During detection, we first obtain the responses of all part filters and cropped root filters in the feature pyramid, before they are fed with the trained weights. The weighed responses of the root and part filters of specific partial configuration are combined to yield a final score pyramid for each PC. Different from training, we will then only use the weights of partial configuration to reweigh the responses of PC, rather than forming new results for PCs. Finally the bounding boxes of partial configurations, together with their scores, are predicted as in [15].

These partial configuration hypotheses will be pre-filtered by a Non-Maximum Suppression (NMS) to ensure that there is only one type of partial configuration in the local area. The following clustering method is the same as PCM. These hypotheses are firstly grouped. In each group, the hypotheses are converted to full-object hypotheses using the spatial interrelationship estimated in Section 3, which is followed by a weighted continuous NMS method of PCM to obtain the final object bounding box hypotheses.

The entire detection process at one scale is illustrated in Figure 8.

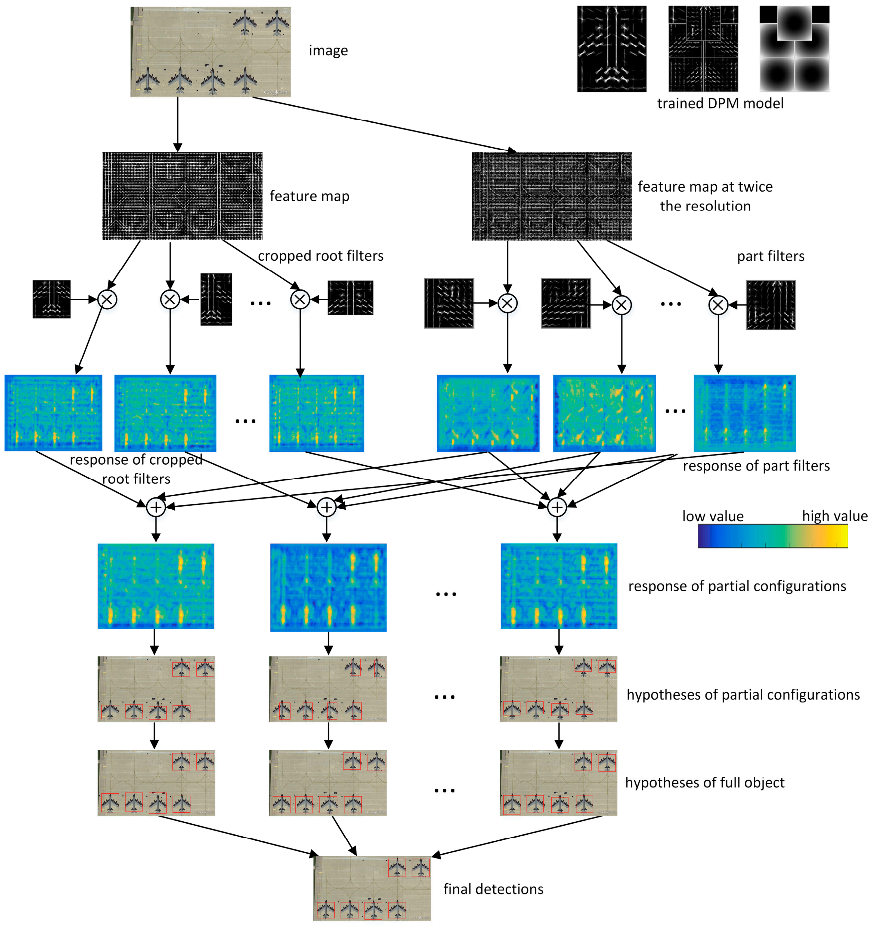


image
trained DPM model
feature map
feature map at twice the resolution
cropped root filters
part filters
...
response of cropped root filters
response of part filters
low value
high value
response of partial configurations
hypotheses of partial configurations
hypotheses of full object
final detections

Figure 8. The entire detection process at one scale. Responses from cropped root and part filters are computed at different resolutions, which are then combined into the responses of partial configurations according to their design. The deformation of part filters is penalized into their corresponding partial configuration. The calculated hypotheses of partial configurations are then converted into that of full object before clustering for final detections.

4. Experimental Results and Discussion

In this section, we provide experimental results and analysis of the proposed UniPCM framework on HR-RSIs. The method is verified on the categories of airplane and ship.

4.1. Data Set Description

For fair comparison with PCM and AFI-PCM, we adopt the same training, validation and test data sets. Their detailed information is provided as follows.

For airplanes, two test sets are used. The first one is the fully visible NWPU10 VHR dataset [20], with a ground sampling distance (GSD) of 0.5–2 m. Its airplane class is chosen, which contains 89 images with 753 airplanes. The other is the more challenging Occlusion dataset [17] which is collected from Google Earth service. There are 47 images with 184 airplanes, with GSD ranging from 0.3–0.6 m. Many airplanes in this dataset are occluded by cloud or hangar, or truncated by image border. The training and validation datasets are collected from the similar area with similar GSD. There are, in total, 2810 samples for training and 282 samples for validation. To be noted, no occluded samples are used for training in our framework.

For ships, one occlusion dataset [17] (hereinafter cited as Ship dataset) is used which is also collected from Google Earth service with a GSD of 0.5–2.4 m. There are, in total, 30 images with 227 ships, 22 objects of which are occluded. The training dataset contains 231 positive samples and 2422 negative samples, and the validation dataset includes 134 samples collected from similar areas.

4.2. Experimental Setup and Evaluation Criteria

To account for in-plane rotation of objects, we manually spilt the full direction into 16 bins, that is, 16 models are used for full-direction detection. The north direction is set as 0. To deal with possible object truncation, we zero-pad the images with 200 pixels for airplanes and 100 pixels for ships. The scale problem of detection is tackled by the inherent multi-scale capacity of DPM through feature pyramid matching. There are 5 parts for both the original DPM models of airplanes and ships. The Intersection-over-Union (IoU) parameter of NMS is 0.3, as calculated in Equation (10). The experiments are conducted on Ubuntu 14.04 LTS with a 4 GHz CPU, a 32 GB RAM, and a NVIDIA Titan X GPU. All codes are implemented with MATLAB.

$$IoU(a,b) = \frac{a \cap b}{a \cup b} \tag{10}$$

The detection results are evaluated according to the criteria of [31]. True positive (TP) refers to a result maintaining an IoU with the ground truth exceeding 0.6, otherwise it is referred as false positive (FP). The unfound ground truths are false negatives (FNs) and the rests are true negatives (TNs). The detection Precision and Recall are defined as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{11}$$

The precision-recall curve (PRC) is used to plot the tradeoff between recall and precision. Average precision (AP) is defined as the area under PRC. Higher AP indicates better performance, and vice versa. As different thresholds yield different precisions and recalls, we use another index, F_1 score, to find the optimal threshold for final detection results. The highest F_1 score makes the best tradeoff between recall and precision, as calculated in Equation (12). A higher optimal F_1 generally means better performance.

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

4.3. Detection Results

In this section, we evaluate the performance of the proposed UniPCM detection framework on the three test sets. PCM [17], AFI-PCM [13], and DPM [15,32] are taken as our comparison baselines. Their parameters are chosen according to optimal trails. The DPM model trained for baseline [15,32] is then used as for partial configuration design for AFI-PCM and UniPCM. Additionally, a deep learning based object detection method, MSCNN [33], is also used in our experiments as another baseline. MSCNN is a multi-scale version of the famous Faster RCNN [34] detection method, which uses feature maps from multiple convolutional layers to enhance the multi-scale detection capability of CNN. Therefore, MSCNN is very suitable for object detection in HR-RSIs. It is originally trained on the much larger dataset ImageNet [35], and then fine-tuned on our training datasets.

All qualitative detection results are obtained with the threshold that leads to the optimal F_1 value. In the following figures, red rectangles correspond to TPs, blue ones correspond to FPs and green ones are FNs. The predicted object direction is denoted above the rectangles. As MSCNN is unable to be direction-invariant, its results are all marked as 0.

4.3.1. Detection Results on NWPU10 VHR Dataset

We first verify the proposed UniPCM method on the fully visible NWPU10 VHR dataset. Figure 9 gives some qualitative detection results of all methods. As seen, most of the airplanes are successfully detected by our method, though with some FPs and FNs. The FNs mainly come from the small objects that are also missed by its counterparts, which indicates that it is due to the inherent defects of the trained DPM model. In some images, UniPCM detects some FPs that are avoided by its baselines. As observed, the structures and appearances of these FPs are very similar to that of an airplane as indicated from the predicted object angle, for example the FP in the fourth column. MSCNN obtains the best results among all methods as expected.

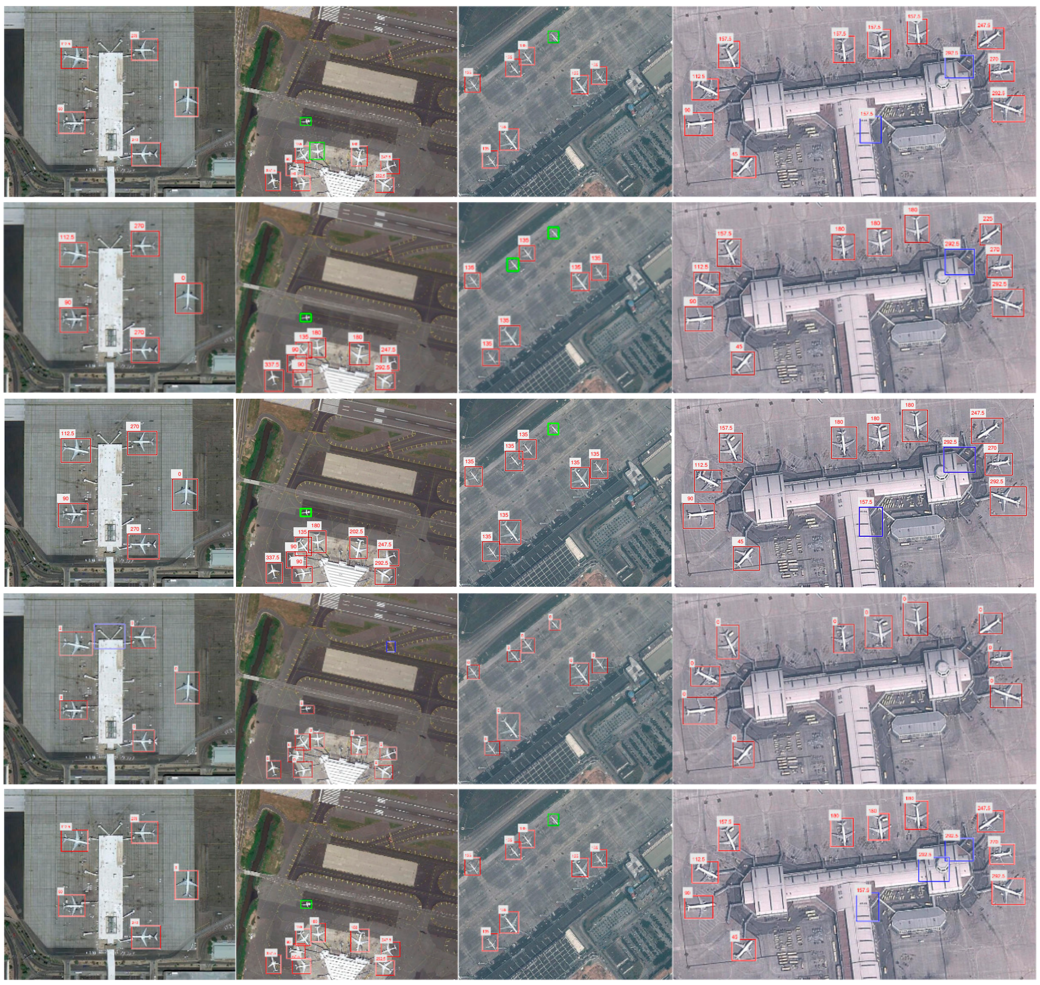


Figure 9. Detection results on NWPU10 VHR dataset of DPM (1st row), PCM (2nd row), AFI-PCM (3rd row), MSCNN (4th row), and UniPCM (5th row).

The quantitative results of PRC and optimal F_1 values are given in Figure 10a and Table 1, respectively. The AP values are denoted in the legend of PRC, which are consistent with the performance difference indicated by optimal F_1 values. It can be observed that UniPCM performs equally to that of PCM and AFI-PCM with very subtle improvement, while outperforming DPM by a large margin. As expected, the deep learning based method, MSCNN, performs best among the methods with an AP up to 0.95. Many airplanes with very small coverage are also successfully detected.

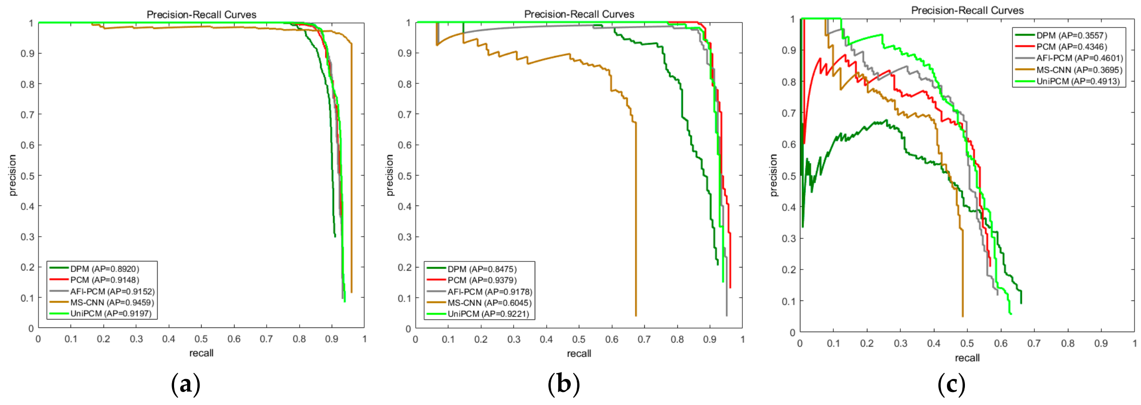


Figure 10. PRC results of all methods on (**a**) NWPU10 VHR dataset; (**b**) Occlusion dataset; and (**c**) Ship dataset.

Table 1. Optimal F_1 values for all methods on all datasets.

Datasets	DPM	PCM	AFI-PCM	MSCNN	UniPCM
NWPU10 VHR dataset	0.889	0.9118	0.9165	0.9481	0.9198
Occlusion dataset	0.8299	0.9314	0.9164	0.6965	0.9231
Ship dataset	0.4764	0.5612	0.5736	0.5042	0.5661

4.3.2. Detection Results on Occlusion Dataset

Figure 11 shows the detection results on the more challenging Occlusion dataset. Despite the fact that partial evidences of objects are missing, the PCM based methods successfully find most of the partial occluded airplanes. UniPCM and AFI-PCM both obtain a few more FNs and FPs, compared to the original PCM. Similarly, some FPs of PCM are avoided by UniPCM and AFI-PCM. Unsurprisingly, the performance of MSCNN decreases greatly compared with that on the NWPU10 VHR dataset. Its capability of discovering partial occluded objects is weak as they are regarded as background with insufficient full-object evidence.

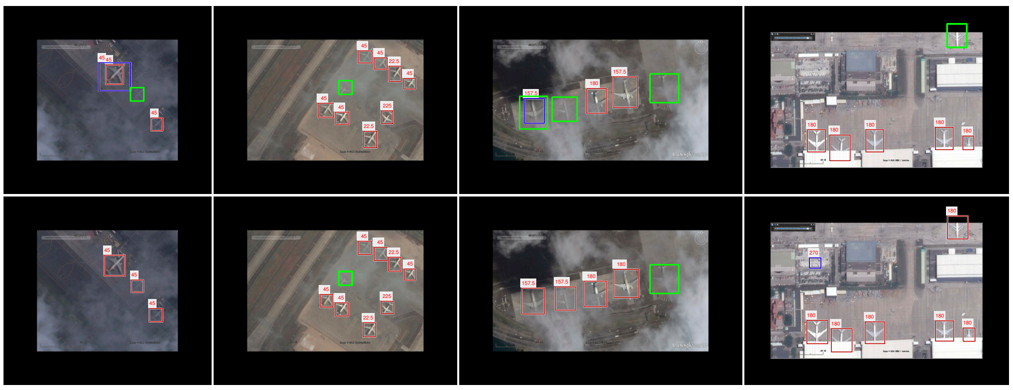


Figure 11. *Cont.*

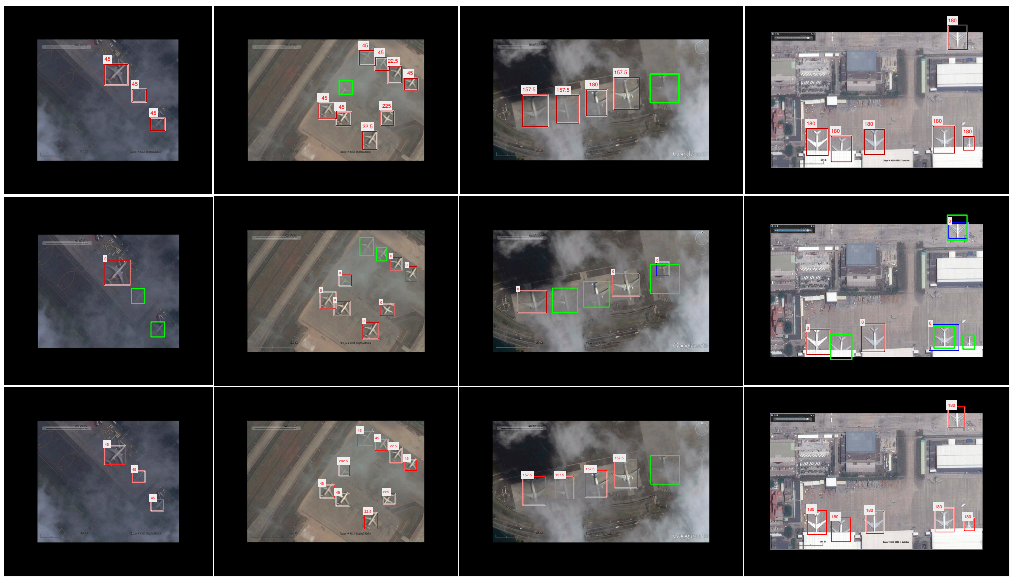


Figure 11. Detection results on Occlusion dataset of DPM (1st row), PCM (2nd row), AFI-PCM (3rd row), MSCNN (4th row), and UniPCM (5th row).

Their performances are also reflected in their PRC and F_1 values in Figure 10b. The APs and F_1 of UniPCM and AFI-PCM are very close and slightly worse than PCM, but outperform MSCNN and DPM by a large margin. Notably, the early decline of AFI-PCM in PRC does not appear in UniPCM, which indicates that its weights are more reasonable.

4.3.3. Detection Results on Ship Dataset

As ships vary from length and size, and many of them are in harbors, ship detection has always been a very difficult task [36]. The detection results on the Ship dataset of all methods are given in Figure 12. As can be seen, all of the methods obtain a number of FNs and FPs, including the deep learning method MSCNN. Despite the fact that their performances are relatively worse than the airplane datasets, many occluded objects are successfully detected. The ship occluded by cloud in the second column missed by DPM, PCM, and AFI-PCM is grasped by MSCNN and UniPCM. However, MSCNN detects more visible ships and maintains a relatively worse object localization accuracy.

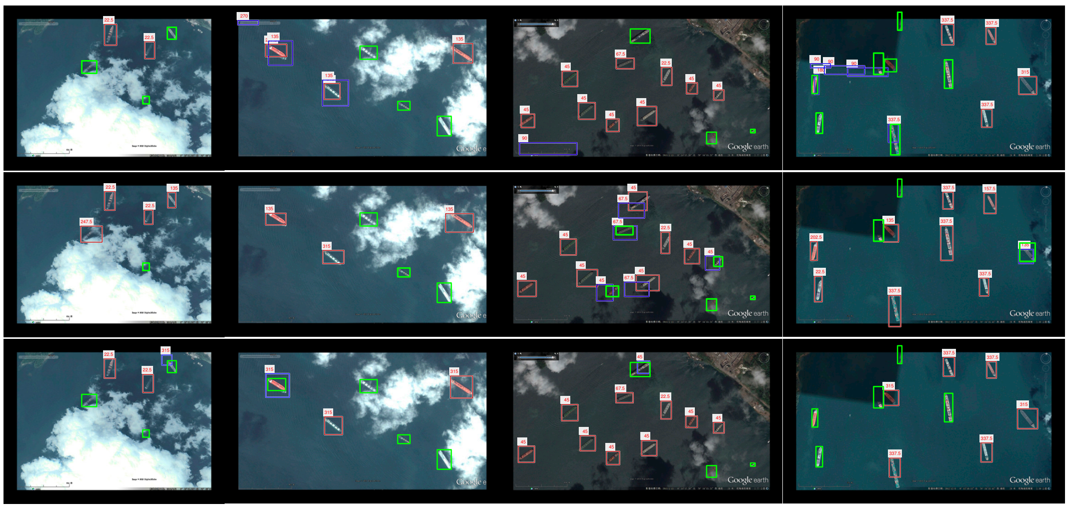


Figure 12. *Cont.*

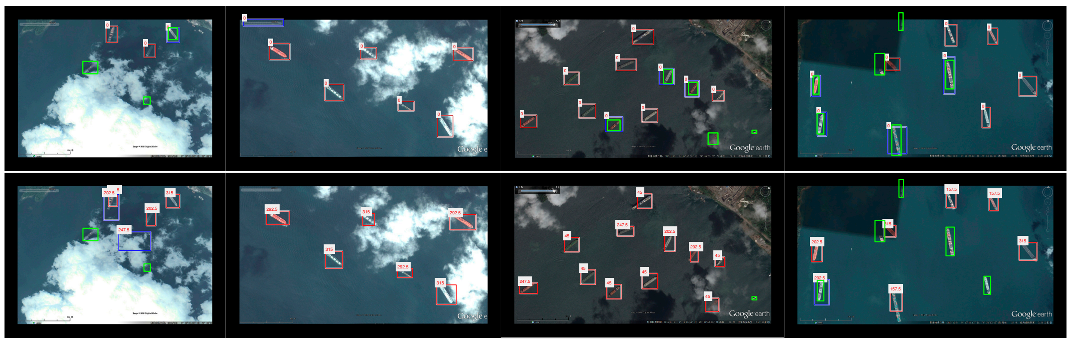


Figure 12. Detection results on Ship dataset of DPM (1st row), PCM (2nd row), AFI-PCM (3rd row), MSCNN (4th row), and UniPCM (5th row).

Their quantitative results PRC and optimal F_1 values are given in Figure 10c and Table 1. UniPCM achieves the highest AP value among all baselines, which is nearly 0.03 and 0.06 larger than AFI-PCM and PCM. UniPCM obtains the best precision when recall is about 0.15 to 0.4. It partially proves that its scoring mechanism is better than other PCM based methods. Optimal F_1 value is coincident with PRC. AFI-PCM maintains a relatively higher F_1 than UniPCM, while PCM performs worse than these two.

4.4. Scheme Analysis

In this section, we analyze the detection performance of the proposed UniPCM framework under different scheme conditions on the NWPU10 VHR dataset and the Occlusion dataset. The following two factors that will influence the final results are analyzed.

4.4.1. Different Spatial Interrelationship Estimation Schemes

In this paper, we propose to directly estimate the spatial interrelationship from the deformation information of the trained DPM. In AFI-PCM, this relationship is estimated from the re-detection of training samples. We evaluate these two different schemes in our experiments. Their PRC results are illustrated in Figure 13 with the same experimental configuration.

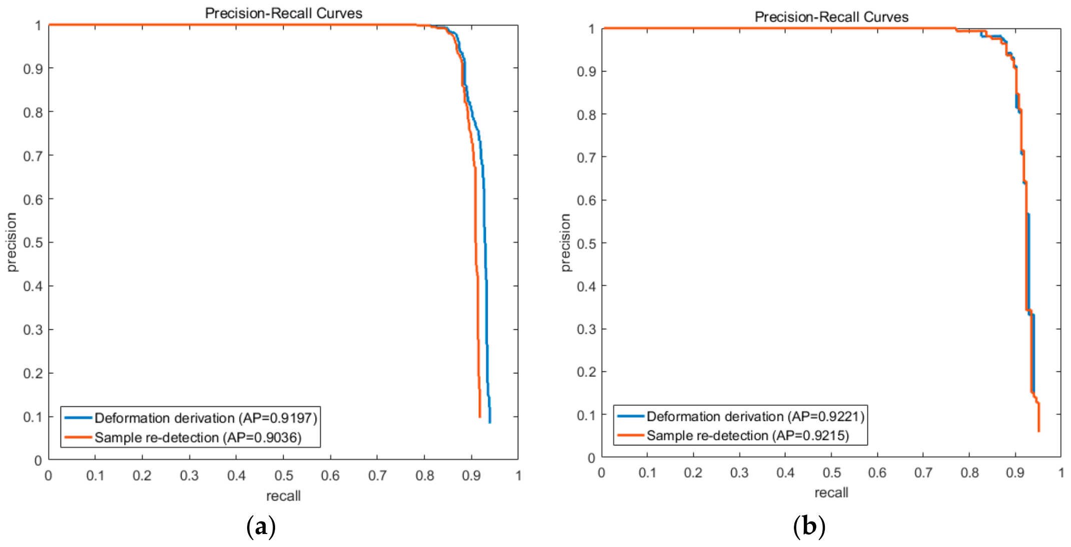


Figure 13. PRC results of different spatial interrelationship estimation schemes on (**a**) NWPU10 VHR dataset and (**b**) Occlusion dataset.

As illustrated, their performances on the Occlusion dataset are very close to each other, and the deformation derivation scheme is a little better than sample re-detection on the NWPU10 VHR dataset. It is mainly because the sample re-detection process maintains uncertainty during detection, since the part detection is a process of global optimization of all possible locations and it can be affected by varied samples. The results indicate that the deformation derived spatial interrelationship can achieve a slightly better performance with much less computation during estimation.

4.4.2. Different Weights Learning Schemes

AFI-PCM uses two different datasets to learn independent intra-PC and inter-PC weights, while UniPCM integrates them into one step on only one dataset. To evaluate this difference, we conduct experiments on the independent weights learning scheme and compare them with the results of UniPCM. Figure 14 shows their PRC results.

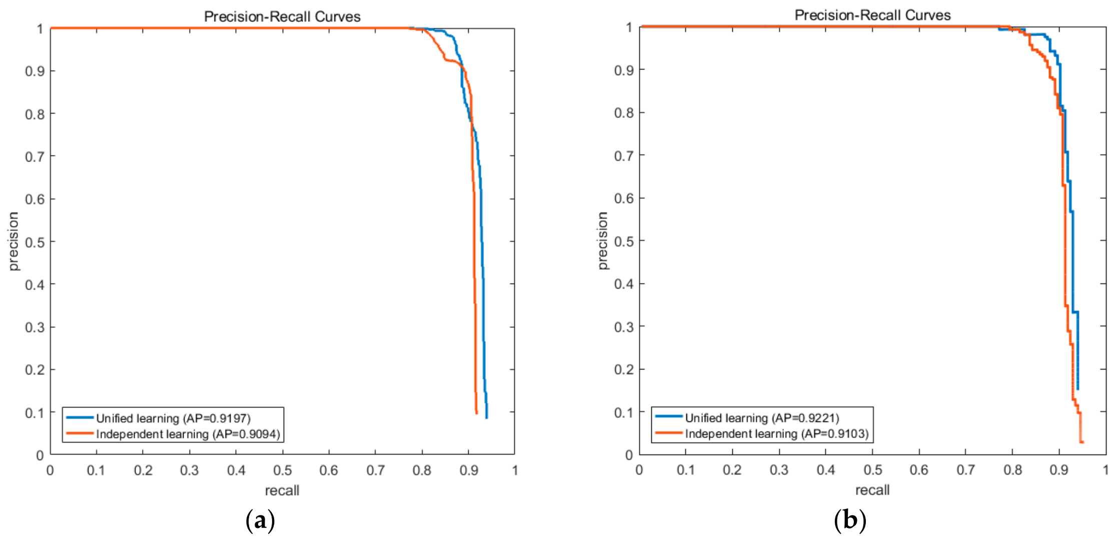


Figure 14. PRC results of different weights learning schemes on (**a**) NWPU10 VHR dataset and (**b**) Occlusion dataset.

Both results on the NWPU10 VHR dataset and the Occlusion dataset illustrate that the unified weights learning process is better than the independent one. On the NWPU10 VHR dataset, UniPCM gets an AP improvement of 0.01 which gets to 0.012 on the Occlusion dataset. It is noticeable that independent learning gets a relatively higher precision when recall is about 0.9, but soon declines dramatically when recall goes up. The results on two datasets verify that unified weights learning can achieve better balance between the elements in partial configurations and also between partial configurations, and is thus better for unified detection than independent learning.

4.5. Training Time Analysis

In this section, we analyze the training time consumed by all baselines. As the architecture of MSCNN is totally different from others, we only analyze the training overhead of DPM, PCM, AFI-PCM, and UniPCM in detail. MSCNN runs on the GPU and takes about 4.5 h for training.

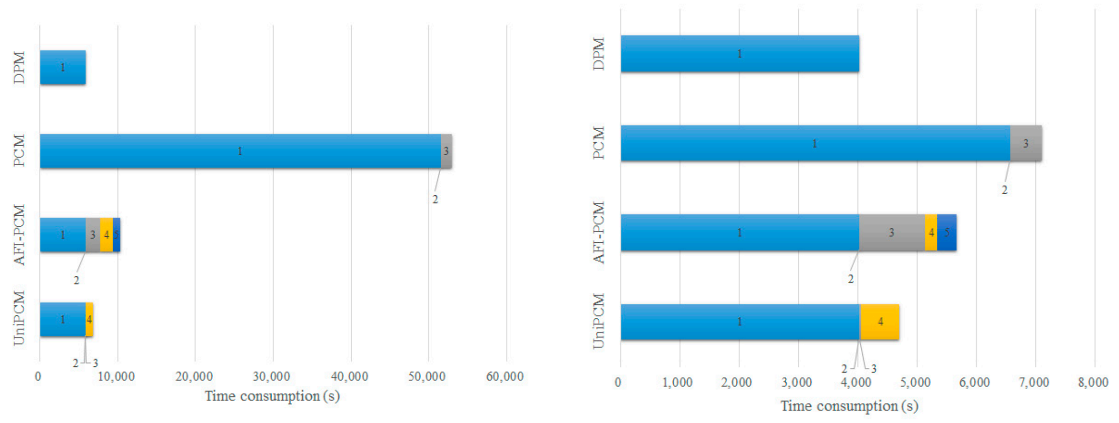
There are two groups of work for PCM training for all these methods. The first one is the manual work that mainly includes sample annotation and category predefinition. For PCM, it contains semantic part definition, category skeleton graph definition, and part-level sample annotation. For AFI-PCM and UniPCM, they only include object-level sample annotation. As manual work is hard to evaluate quantitatively, we do not analyze it in this section. As for manual work, UniPCM and AFI-PCM have already greatly reduced the workload. The annotation work of UniPCM and AFI-PCM is the same with DPM.

The other work for PCM training is the model training. To be brief, we list all related training stages of these three methods in Table 2.

Table 2. Training stages of different methods, where 1 means running one time, *N* corresponds to N times, √ indicates that it is included, and - refers to absence in the method.

Stage No.	Training Stages	DPM	PCM	AFI-PCM	UniPCM
1	DPM model training	1	*N*	1	1
2	partial configuration design	-	-	√	√
3	spatial interrelationship estimation	-	√	√	√
4	intra-PC weights learning	-	-	√	√
5	inter-PC weights learning	-	√	√	

In this section, we only take the north direction of airplane and ship models as our experimental subjects. The statistical results of the model training stage of these methods are illustrated in the bar chart of Figure 15. The overheads of training one DPM model for airplane and ship categories are 5844 s and 4022 s, respectively. Note that DPM has much lower detection accuracy compared with PCM based methods, although it achieves shorter time overheads since it is only the first training step of PCM based methods. Compared with PCM, AFI-PCM largely reduces the training volume to about 19% and 80% for the two categories, that is, a speedup up to 5.16× and 1.25×. The majority of saved time is the multi-DPM model training of PCM. Compared to AFI-PCM, UniPCM cuts down the training time by another 40% and 17%. Taking the airplane category as an example, step 3 of UniPCM is done within 15 s, while AFI-PCM takes about 1923 s. The step of weights learning, including scores preparation, is also decreased to about 983 s, while AFI-PCM takes about 2501 s. A similar performance acceleration can be also observed on the ship model training.



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

Figure 15. Training time consumption of categories of (**a**) airplane and (**b**) ship, with the numbers above indicating its stage number in Table 2.

As the numbers of partial configurations in PCM and UniPCM are different, we take a second look at the speedup obtained by our UniPCM. To generate one partial configuration for airplanes and ships, AFI-PCM needs 15% and 48% of the time consumed by PCM, while UniPCM only uses nearly 10% and 40% of PCM, that is, a speedup of 10× and 2.5×, respectively. We will achieve greater speedup if more partial configurations are designed. This speedup for training is obvious, and it is convenient for the fast implementation to new categories of objects, let alone the time already saved for massive manual annotation work.

4.6. Detection Time Analysis

AFI-PCM and PCM simply detect N DPM models of partial configurations, that is, it is nearly N times of one single DPM detection. UniPCM only convolves through the image once and then directly obtains the responses of all partial configurations. To evaluate the more important detection speed, we obtain the statistical time consumption of PCM, AFI-PCM, and UniPCM for full-direction detection. Figure 16 illustrates the time consumption per partial configuration per image of the three methods. The average image sizes of datasets are 683×970, 1125×1457, and 987×1747. Similar to Section 4.5, we give the detection time consumption of MSCNN of each image in the three datasets, which is about 0.15 s, 0.32 s, and 0.35 s, respectively.

Since a full-object DPM model generally has more filters than a single partial configuration, it takes the highest time consumption among all baselines as in Figure 2, which partially supports that obtaining the filter responses takes great detection time. As indicated in Figure 2, nearly three quarters of AFI-PCM and PCM detection computation involves calculating for the responses of filters. Although AFI-PCM shares the parts between partial configurations, their filter responses are actually wasted and unshared during detection. UniPCM makes full use of these responses and its total time consumption for one partial configuration is only 0.2 s for NWPU10 VHR, 0.4 s for Occlusion, and one second for Ship dataset; this is almost only 14%, 24%, and 40% of AFI-PCM, respectively, that is, a speedup up to $7.2\times$, $4.1\times$ and $2.5\times$ over AFI-PCM, and $5.1\times$, $3.7\times$ and $2\times$ over PCM. UniPCM makes fast POOD become possible. The detection speeds of AFI-PCM and PCM are roughly the same, with differences mainly coming from the extra weights multiplication in AFI-PCM.

To achieve full-direction detection of objects in the experiment, AFI-PCM and UniPCM maintain 208 and 188 partial configurations for the categories of airplane and ship, while PCM uses 160 and 48 partial configurations. It can also be inferred from Figure 16 that when more partial configurations are designed, UniPCM will obtain a much higher detection speedup.

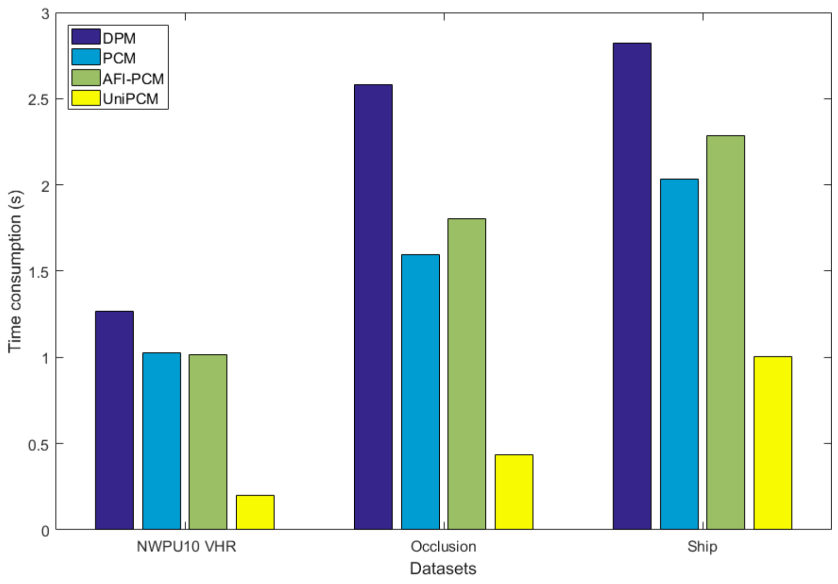


Figure 16. Detection time consumption per partial configuration per image of PCM based methods. Specifically, the results of DPM are the time consumption per model per image.

5. Conclusions

Partial occluded object detection (POOD) has become a very serious problem for applications that use high-resolution remote sensing images. The recently published PCM deals with this problem with relatively heavy computation and a massive labor workload. Its improved variant, AFI-PCM, also shares similar shortcomings of slow detection. To tackle this, a unified PCM framework (UniPCM) is proposed in this paper for fast POOD. Based on an idea that shares the same set of parts for all automatically designed partial configurations, the proposed method successfully integrates the entire training and detection pipeline into a unified one. Additionally, a fast and direct spatial interrelationship estimation method and a unified weights learning approach are proposed. Experimental results on three datasets indicate that the proposed UniPCM method achieves a comparable accuracy, while obtaining a much faster model training and image detection speed compared to PCM and AFI-PCM. It makes fast POOD become possible. In the future, we intend to make this unified POOD framework into a neural network fashion.

Acknowledgments: The work is supported in part by the Program for New Century Excellent Talents in the University of China under Grants NCET-11-0866 and in part by the National Natural Science Foundation of China under Grants 41601487.

Author Contributions: Shaohua Qiu and Gongjian Wen contributed to the idea and the data collection of this study. Shaohua Qiu, Jia Liu, Zhipeng Deng and Yaxiang Fan developed the algorithm, performed the experiments, analyzed the experimental results, and wrote this paper. Gongjian Wen, Jia Liu, Zhipeng Deng and Yaxiang Fan supervised the study and reviewed this paper.

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

Appendix A

Appendix A.1. PDF Estimation of Max/Min Functions

For $\xi = \max\{\xi_1, \xi_2, \cdots, \xi_n\}$, we have $\{\xi > x\} \subseteq \cup_{i=1}^{n}\{\xi_i > x\}$, then the probability of $\xi > x$ is

$$P(\xi > x) = 1 - \prod_{i=1}^{n} P(\xi_i \le x)$$

$$P(\xi \le x) = \prod_{i=1}^{n} P(\xi_i \le x)$$

We convert it to the formulation of Cumulative Distribution Function (CDF):

$$F(z) = \prod_{i=1}^{n} F(z_i)$$

Similarly, the CDF of $\xi = \min\{\xi_1, \xi_2, \cdots, \xi_n\}$ is

$$F(z) = 1 - \prod_{i=1}^{n}(1 - F(z_i))$$

Thus, the PDF of ξ can be obtained by calculating PDF of CDF $F(z)$.

Appendix A.2. PDF Estimation of Sum/Subtract/Divide Functions

For $Z = X + Y$ and their corresponding PDF $f_X(x), f_Y(y)$, we have

$$F_Z(z) = P\{Z \le z\} = \iint\limits_{x+y \le z} f(x)f(y)dxdy$$

Let $x = u - y$, then

$$\begin{aligned} F_Z(z) &= \int_{-\infty}^{+\infty}\left(\int_{-\infty}^{z-y} f(x)f(y)dx\right)dy = \int_{-\infty}^{+\infty}\left(\int_{-\infty}^{z} f(u-y)f(y)du\right)dy \\ &= \int_{-\infty}^{z}\left(\int_{-\infty}^{+\infty} f(u-y)f(y)dy\right)du \end{aligned}$$

Therefore, the PDF of Z is

$$f_{X+Y}(z) = \int_{-\infty}^{+\infty} f_X(z-y) f_Y(y) dy = \int_{-\infty}^{+\infty} f_X(x) f_Y(z-x) dx$$

Similarly, the PDF of $Z = X - Y$ is

$$f_{X-Y}(z) = \int_{-\infty}^{+\infty} f_X(z+y) f_Y(y) dy = \int_{-\infty}^{+\infty} f_X(x) f_Y(x-z) dx$$

The PDF of $Z = X/Y$ is

$$f_{X/Y}(z) = \int_{-\infty}^{+\infty} |x| f_X(xz) f_Y(x) dx$$

References

1. Han, X.; Zhong, Y.; Zhang, L. An efficient and robust integrated geospatial object detection framework for high spatial resolution remote sensing imagery. *Remote Sens.* **2017**, *9*, 666. [CrossRef]
2. Chen, X.; Xiang, S.; Liu, C.-L.; Pan, C.-H. Vehicle detection in satellite images by hybrid deep convolutional neural networks. *IEEE Geosci. Remote Sens. Lett.* **2014**, *11*, 1797–1801. [CrossRef]
3. Cai, B.; Jiang, Z.; Zhang, H.; Zhao, D.; Yao, Y. Airport detection using end-to-end convolutional neural network with hard example mining. *Remote Sens.* **2017**, *9*, 1198. [CrossRef]
4. Li, X.; Wang, S. Object detection using convolutional neural networks in a coarse-to-fine manner. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 2037–2041. [CrossRef]
5. Ševo, I.; Avramović, A. Convolutional neural network based automatic object detection on aerial images. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 740–744. [CrossRef]
6. Deng, Z.; Sun, H.; Zhou, S.; Zhao, J.; Zou, H. Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2017**, *PP*, 1–13. [CrossRef]
7. Cheng, G.; Zhou, P.; Han, J. Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 7405. [CrossRef]
8. Long, Y.; Gong, Y.; Xiao, Z.; Liu, Q. Accurate object localization in remote sensing images based on convolutional neural networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2486–2498. [CrossRef]
9. Li, H.; Fu, K.; Yan, M.; Sun, X.; Sun, H.; Diao, W. Vehicle detection in remote sensing images using denoizing-based convolutional neural networks. *Remote Sens. Lett.* **2017**, *8*, 262–270. [CrossRef]
10. Tang, T.; Zhou, S.; Deng, Z.; Lei, L.; Zou, H. Arbitrary-oriented vehicle detection in aerial imagery with single convolutional neural networks. *Remote Sens.* **2017**, *9*, 1170. [CrossRef]
11. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. *arXiv* **2017**, arXiv:1703.06211.
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet Classification with Deep Convolutional Neural Networks. In Proceedings of the NIPS'12, 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
13. Qiu, S.; Wen, G.; Deng, Z.; Fan, Y.; Hui, B. Automatic and fast pcm generation for occluded object detection in high-resolution remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 1730–1734. [CrossRef]
14. 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. [CrossRef]
15. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1627–1645. [CrossRef] [PubMed]
16. Cheng, G.; Han, J.; Guo, L.; Qian, X.; Zhou, P.; Yao, X.; Hu, X. Object detection in remote sensing imagery using a discriminatively trained mixture model. *ISPRS J. Photogramm. Remote Sens.* **2013**, *85*, 32–43. [CrossRef]
17. Qiu, S.; Wen, G.; Fan, Y. Occluded object detection in high-resolution remote sensing images using partial configuration object model. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2017**, *10*, 1909–1925. [CrossRef]
18. Wang, G.; Wang, X.; Fan, B.; Pan, C. Feature extraction by rotation-invariant matrix representation for object detection in aerial image. *IEEE Geosci. Remote Sens. Lett.* **2017**, *PP*, 1–5. [CrossRef]

19. Zhao, A.; Fu, K.; Sun, H.; Sun, X.; Li, F.; Zhang, D.; Wang, H. An effective method based on acf for aircraft detection in remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 744–748. [CrossRef]
20. Cheng, G.; Han, J.; Zhou, P.; Guo, L. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS J. Photogramm. Remote Sens.* **2014**, *98*, 119–132. [CrossRef]
21. Mountrakis, G.; Im, J.; Ogole, C. Support vector machines in remote sensing: A review. *ISPRS J. Photogramm. Remote Sens.* **2011**, *66*, 247–259. [CrossRef]
22. Lei, Z.; Fang, T.; Huo, H.; Li, D. Rotation-invariant object detection of remotely sensed images based on texton forest and hough voting. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 1206–1217. [CrossRef]
23. Zhang, W.; Sun, X.; Wang, H.; Fu, K. A generic discriminative part-based model for geospatial object detection in optical remote sensing images. *ISPRS J. Photogramm. Remote Sens.* **2015**, *99*, 30–44. [CrossRef]
24. Bi, F.; Chen, J.; Zhuang, Y.; Bian, M.; Zhang, Q. A decision mixture model-based method for inshore ship detection using high-resolution remote sensing images. *Sensors* **2017**, *17*, 1470. [CrossRef] [PubMed]
25. Zia, M.; Stark, M.; Schindler, K. Explicit occlusion modeling for 3d object class representations. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, OR, USA, 23–28 June 2013; pp. 3326–3333.
26. Ouyang, W.; Wang, X. A discriminative deep model for pedestrian detection with occlusion handling. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3258–3265.
27. Niknejad, H.T.; Kawano, T.; Oishi, Y.; Mita, S. Occlusion handling using discriminative model of trained part templates and conditional random field. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Gold Coast, QLD, Australia, 23–26 June 2013; pp. 750–755.
28. Lee, D.T.; Schachter, B.J. Two algorithms for constructing a delaunay triangulation. *Int. J. Comput. Inf. Sci.* **1980**, *9*, 219–242. [CrossRef]
29. West, D.B. *Introduction to Graph Theory*; Prentice Hall: Upper Saddle River, NJ, USA, 2001; Volume 2.
30. Grant, M.; Boyd, S. Cvx: Matlab Software for Disciplined Convex Programming, Version 2.0 Beta. 2013. Available online: http://cvxr.com/cvx (accessed on 14 March 2018).
31. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
32. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D. Discriminatively Trained Deformable Part Models, Release 4. Available online: http://people.cs.uchicago.edu/~pff/latent-release4/ (accessed on 14 March 2018).
33. Cai, Z.; Fan, Q.; Feris, R.; Vasconcelos, N. A unified multi-scale deep convolutional neural network for fast object detection. In Proceedings of the European Conference on Computer Vision 2016, Amsterdam, The Netherlands, 8–16 October 2016; pp. 354–370.
34. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
35. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Li, F.F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
36. Xu, F.; Liu, J.; Dong, C.; Wang, X. Ship detection in optical remote sensing images based on wavelet transform and multi-level false alarm identification. *Remote Sens.* **2017**, *9*, 985. [CrossRef]