IPSJ Transactions on Computer Vision and Applications Vol.7 10–17 (Jan. 2015)

[DOI: 10.2197/ipsjtcva.7.10]

Research Paper

Aircraft Detection by Deep Convolutional Neural Networks

Xueyun Chen[1,a)] Shiming Xiang[2,b)] Cheng-Lin Liu[2,c)] Chun-Hong Pan[2,d)]

Received: February 17, 2014, Accepted: October 20, 2014, Released: January 28, 2015

Abstract: Features play crucial role in the performance of classifier for object detection from high-resolution remote sensing images. In this paper, we implemented two types of deep learning methods, deep convolutional neural network (DNN) and deep belief net (DBN), comparing their performances with that of the traditional methods (handcrafted features with a shallow classifier) in the task of aircraft detection. These methods learn robust features from a large set of training samples to obtain a better performance. The depth of their layers (>6 layers) grants them the ability to extract stable and large-scale features from the image. Our experiments show both deep learning methods reduce at least 40% of the false alarm rate of the traditional methods (HOG, LBP+SVM), and DNN performs a little better than DBN. We also fed some multi-preprocessed images simultaneously to one DNN model, and found that such a practice helps to improve the performance of the model obviously with no extra-computing burden adding.

Keywords: remote sensing, object detection, deep belief nets, deep convolutional neural networks

1. Introduction

Aircraft detection is an important task for both military and commercial applications. One thinks it might have been solved well. For more than 10 years, lots of work has been done [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12] on detection of different types of small objects from large remote sensing images, such as aircraft and vehicle. Many methods depending on the combinations of various features have been proposed.

Yet the fact is aircraft detection remains an unsolved challenge, no really satisfactory result has been made of aircraft detection in a large set of complex real airports images, no locating method has been found efficient enough to locate them quickly from large images (20000×20000, for instance), no feature has been proved robust enough to overcome the influence of various illumination.

In the past literatures, Cai et al. [4] showed the difficulty to segment aircraft exactly from its backgrounds by the effect of shadow. They used an anisotropic heat diffusion model to remove the shadow. However, their method only worked well for white aircrafts, more likely, failed in the cases of aircrafts with various colors. Global thresholding method has been proved efficient in removing the background of white aircrafts [1], [3]. **Figures 1** and **2** show that image thresholding at a suitable value shows a better effect than the gradient or the canny edge images, and locating white aircraft on thresholding images is easier than that on gradient images.

However, **Fig. 3** shows that some blue aircrafts have disappeared, and no suitable thresholding can separate them from their background, because the blue color and the background have an equal gray scale. But they can be located successfully on a gradient image. The method based on the gradient image can not dis-

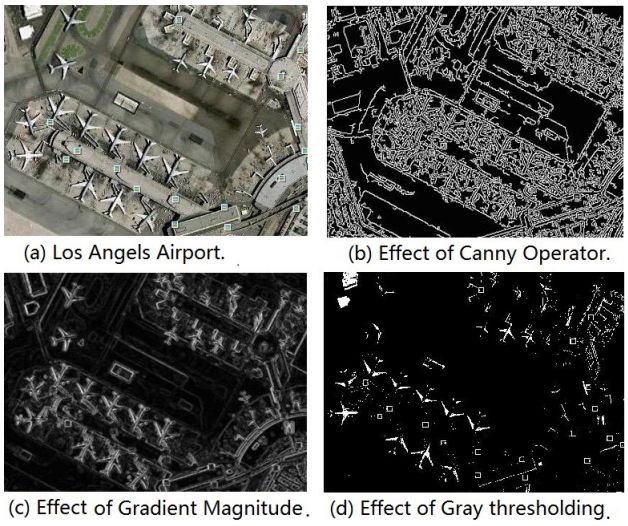

Fig. 1 Locating white aircrafts on the gray thresholding image (d) is easier than that on the image of canny (b) and gradient image of Dalal and Triggs [15] (c).

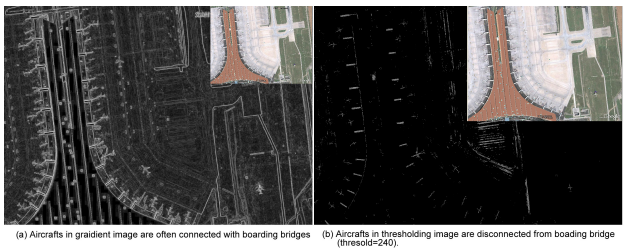

Fig. 2 On a gradient image, some aircrafts are connected with their boarding bridges, this adds difficulty to the locating problem. Suitable gray thresholding separates the white aircrafts from such attachments of background, makes the locating problem easier.

1 College of Electrical Engineering, Guangxi University, Nanning 530007, China
2 National Laboratory of Pattern Recognition, Institute of Automation, Chinese academy of Sciences, Beijing 100190, China
a) xueyun.chen@nlpr.ia.ac.cn
b) smxiang@nlpr.ia.ac.cn
c) liucl@nlpr.ia.ac.cn
d) chpan@nlpr.ia.ac.cn

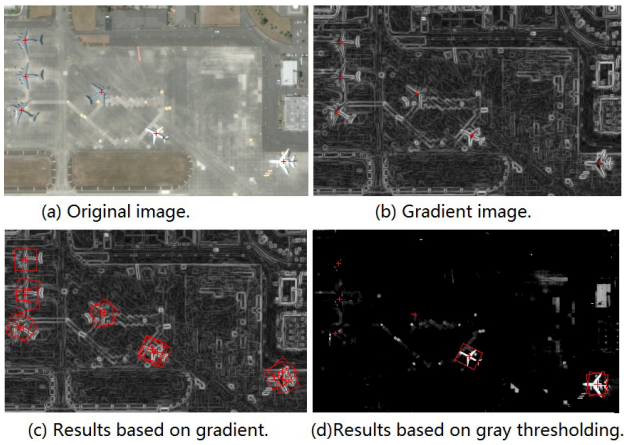

Fig. 3 Locating colorful aircrafts on gradient image is easier than that on gray thresholding image.

tinguish the edge of objects from their shadows, and this indeed reduces its efficiency to some degree, but its partial immunity to changes of color and illumination is worth appreciating. Dalal and Triggs [15] proposed a simple gradient computing method by the maximal norm of the three RGB color channels. Figure 1 shows it performs much better than canny algorithm. In the paper, we utilize Dalal and Triggs' gradient to locate colorful aircrafts.

Features are critical for the performance of object detectors, combinations of different types of features have been tested for object detection from satellite images. Hsieh [1] used aircraft contour, Zernike moments, wavelet and SVM classifier to detect aircraft. Yildiz and Polat [2] used Gabor+SVM. Liu et al. [3] proposed a coarse-to-fine shape modeling method based on edge computing (Sobel). Sun et al. [5] used the key-points and spatial sparse coding bag-of-words model to detect aircraft. Li et al. [6] used visual saliency computation and symmetry detection. Tien et al. [8] used cross-ratios to model curve data of aircraft contour. Xu and Duan [9] used artificial bee colony algorithm with edge potential function to seek aircraft targets. However, invariant moment, saliency and symmetry features, geometric contour, edge, shape and curve data are not stable to the disturbance of all kinds, especially for tiny blurred aircraft. Background and illumination will impose considerable influences on such geometrical features. Grabner et al. [14] used boosting method based on Haar wavelets, HOG (histogram of oriented gradients) [15] and LBP (local binary patterns) [32]. Kembhavi et al. [11] computed multi-scales HOG features on color maps to detect vehicle in the San Francisco images from google earth, they showed that HOG outperforms SIFT (Scale Invariant Feature Transform) [33].

SIFT, LBP, HOG and Gabor [31] are popular features used in object detection. Both of SIFT and HOG rely on the gradient histogram of blocks, but HOG has a flexible bins of gradient orientation and overlapped blocks of dividing pattern. They are stable because the gradient norm is stable, but the gradient orientation is not as stable as the magnitude, So the orientation bins cannot be cut too fine (usually 8 or 9 bins). LBP is the histogram of binary patterns of all pixels of a block. It would be a good texture descriptor, only if its patterns were dividing into suitable bins, and because it relies on the gray scale, it is not stable to noise and illumination. Gabor is actually a multi-scales (usually 5) and multi-directions (usually 8) gradient descriptor, which is used widely in saliency computation and object recognition. Gabor is not stable because it has no statistic expression like a histogram of something. **Of cause, Gabor can be transformed into a histogram-type descriptor as HOG does, and we believe such a transformation will enhance its stability more likely.** Another problem is the scale variety of objects in real images. In most applications, features are computed on overlapped blocks of variable scales to enhance its scale-invariant capability.

In the case when only a small training set is available, using such handcrafted fixed features is reasonable. But if you have thousands or more samples for each class (such as in the case of aircraft detection), learning intrinsic features from the training samples is more advisable. Such features are now learned by the deep learning methods from the input data automatically. Figures 5 and 11 show these features are random and noisy images, actually, no existent theory has given a satisfactory explanation on why such features work well, the actual roles of such features remains as much a mystery as it was when Hinton first proposed the deep belief nets (DBN) [21] in 2006.

Convolutional neural networks (CNN) originates from Hubel and Wiesel's study [16] on cats striate cortex. They first proposed the concept "receptive field". Fukushima [17] designed a self-organizing neural networks, which was unaffected by shifts of position. The normal structure of CNN was proposed by LeCun et al. [18] who first used the concept "convolutional layer". Garcia an Delakis [19] used a 6-layer CNN for face detection in CMU and MIT test sets. Recently, Ciresan et al. [20] presented the structure of Deep CNN (DNN), and achieved the state-of-the-art performances on six benchmark image classification databases, including the MNIST (handwritten digits), NIST SD-19, handwritten Chinese characters, traffic signs, CIFAR10 and NORB. The results in MNIST and traffic signs are even superior to human performance.

Yu et al. [24] first showed that DBN achieved very promising recognition results on large vocabulary speech recognition tasks. Their work revealed the potential power of deep learning method in practical application. Later, many works were done using DBN for speech recognition [25], [26], [27], [28], [29], [30]. it seems that DNN is more suitable for image classification, and DBN is suitable for speech recognition. In this paper, we compare both types of deep learning methods in aircraft detection, and show that DNN outperforms DBN slightly.

The remainder of this paper is organized as the following: Section 2 presents the architectures of the DNN and DBN. Section 3 gives the implementation details of our algorithm of aircraft detection, we implemented DNN and DBN by ourselves. Section 4 presents the experimental results, and Section 5 makes the conclusion.

2. Deep Learning Methods

In this section, we first discuss the structure of DNN, then we discuss the structure of DBN, and its pretraining process.

2.1 Deep Convolutional Neural Networks

The layers of DNN can be divided into two parts: feature ex-

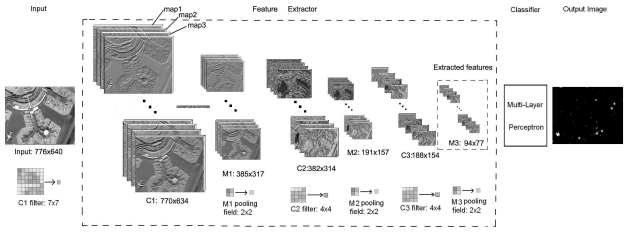

Fig. 4 An example of DNN. Where n_l=3, n_m=84.

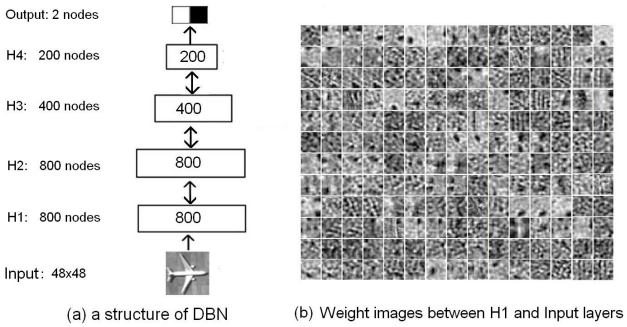

Fig. 5 An example of the structure of DBN and its partial weight images after pretraining.

tractor and Multi-Layer Perceptron (MLP) classifier. Let n_l denote the number of the convolutional layers, n_m denotes the map number of one convolutional layer. For convenience, we suppose all convolutional layers have the same map number. The convolutional layers of DNN are defined as: $C^1, \cdots, C^{n_l}$. The max-pooling layers of DNN are defined as: $M^1, \cdots, M^{n_l}$. All the convolutional and max-pooling layers compose the feature extractor of DNN. M^{n_l} output the extracted features to the MLP Classifier.

MLP classifier includes the hidden layers and the output layer. Its output value can be transformed into the output image (right part of Fig. 1) where bright dots represent the aircraft candidates. The brightness of the dot is proportionate to the classifier output value. The *tanh* function is used as the kernel function for all nodes in DNN. **Figure 4** gives an example of DNN. The convolutional layer maps are determined by the filters sliding on the previous layer pixel by pixel. The max-pooling layer maps are determined by the max-pooling function on the non-overlapped max-pooling fields sliding over the previous convolutional layer. The max-pooling function has two significant effects: reducing the map size, enhancing the shift-invariant ability and anti-noise ability by the "winner-take-all" principle.

2.2 Deep Belief Nets

Deep Belief Nets (DBN) are consisted by a visible input layer, several hidden layers and an output layer. The visible layer input the image data, whose gray range has been normalized into [0,1], the hidden layers are invisible, their state are binary values, being activated by the sigmoid kernel function. **Figure 5** show an example of the structure of DBN.

The Restricted Boltzmann Machine (RBM) is the basic block of Deep Belief Networks (DBN), it is trained by a learning algorithm called Contrastive Divergence (CD) [20], [21], which uses the Gibbs sampling and the reconstruction error to train the weights of RBM. The energy function of RBM is defined by [23]:

$$E(v,h) = -\sum_{\mathrm{ij}} v_{\mathrm{i}} h_{\mathrm{j}} W_{\mathrm{ij}} - \sum_{\mathrm{i}\in\mathrm{pixels}} v_{\mathrm{i}} c_{\mathrm{i}} - \sum_{\mathrm{j}\in\mathrm{hiddenlayer}} h_{\mathrm{j}} b_{\mathrm{j}} \quad (1)$$

where v_{i} is the pixel of the visible input layer, h_{j} is the node of the hidden layer, whose value must be 0 or 1. b_j and c_i are their biases, W_{ij} are the weights of RBM, its update formula is given by:

$$\begin{cases} \Delta W_{ij} = -\varepsilon \frac{\partial E}{\partial W_{ij}} = \varepsilon(\langle v_i h_j \rangle|_v - \langle v_i h_j \rangle|_{recon}) \\ \Delta b_j = -\varepsilon \frac{\partial E}{\partial b_j} = \varepsilon(h_j|_v - h_j|_{recon}) \\ \Delta c_i = -\varepsilon \frac{\partial E}{\partial c_i} = \varepsilon(v_i|_v - v_i|_{recon})) \end{cases} \quad (2)$$

ε is the LearnRate, $\langle\ \rangle$ is the inner product. $*|_v$ means $*$ is get from visible input data. $*|_{recon}$ denote the reconstruction value of $*$, $*|_v$ are shown as the following:

$$\begin{cases} v_i|_v = v_i \\ h_j|_v = Pro(h_j = 1) = sigm(b_j + \sum_i v_i W_{ij}) \end{cases} \quad (3)$$

$Pro(*)$ is the probability of $*$. $sigm$ is the standard *sigmoid* function. Because the states of the hidden layer are invisible binary value, we perform Gibbs sampling to estimate its states. We denote $rand\_value = 1.0 \times rand()/\mathrm{RAND\_MAX}$, $rand\_value$ is a random value in [0,1]. RAND_MAX is a constant of C language. We have:

$$Sample(h_j) = \begin{cases} 1, & if\ Pro(h_j = 1) > rand\_value \\ 0, & otherwise \end{cases} \quad (4)$$

$Sample(*)$ means the Gibbs sample of $*$. Now we reconstruct the visible layer and the hidden layer:

$$\begin{cases} v_i|_{recon} = sigm(c_i + \sum_j W_{ij} Sample(h_j)) \\ h_j|_{recon} = sigm(b_j + \sum_i v_i|_{recon}\ W_{ij}) \end{cases} \quad (5)$$

The weights update formulas can be rewritten as:

$$\begin{cases} \Delta W_{ij} = \varepsilon(v_i sigm(b_j + \sum_i v_i W_{ij}) \\ \qquad - v_i|_{recon}\ sigm(b_j + \sum_i v_i|_{recon}\ W_{ij})) \\ \Delta b_j = \varepsilon(sigm(b_j + \sum_i v_i W_{ij}) \\ \qquad - sigm(b_j + \sum_i v_i|_{recon}\ W_{ij})) \\ \Delta c_i = \varepsilon(v_i - sigm(c_i + \sum_j W_{ij} Sample(h_j))) \end{cases} \quad (6)$$

The RBM must be trained properly when the reconstruction error diminishes to a small value. All weights of DBN must be pre-trained layer-by-layer as the RBM training. After pre-training, the weights of DBN are fine-tuned by the standard back-propagation algorithm and the steepest descent algorithm as the Multi-Layer Perceptron (MLP).

3. Implementation Detail

In this section, we first discuss the thresholding method we used in gray and gradient images, then we present the orientation computing method we used. Thirdly, we show the multi-scale sliding window technique we used. At last, we exhibit the structure and parameters of DNN we used, and discuss its training processes.

3.1 Gray Thresholding and Gradient Thresholding

Aircraft detection is a difficult problem. As shown in Figs. 1 and 2, locating white aircrafts on a suitable gray thresholding

Fig. 6 For white aircrafts, constant multi-thresholds are suitable for segmenting aircrafts from different airport images.

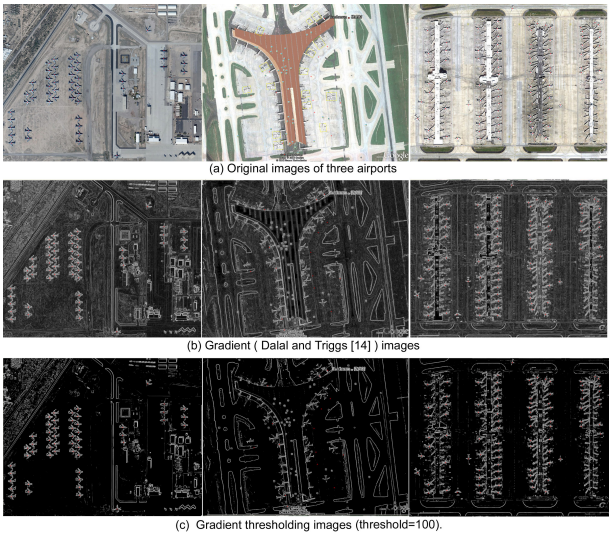

Fig. 7 For aircrafts on the gradient images, suitable thresholding reduces the background textures.

Table 1 Locating results on different preprocessed images.

Preprocessing Images	#samples	locating accuracy(%)
Gray, Gradient	149923	99.15
Gradient	187892	98.15

image is more easy for complex environments. However, computing suitable threshold is a difficult problem, and the risk of an unsuitable threshold is unendurable. So we use multiple constant thresholds. **Figure 6** shows such simple multi-thresholding method is suitable for various airports. It is obvious that the more thresholds are used, the more easier the locating work is. In the experiments of Section 4, three constant thresholds (210, 240, 250) are used. In our database, a few images contain colorful aircrafts which can not be located on gray thresholding images. We locate those colorful aircrafts on their gradient (Dalal and Triggs [15]) images. However, to erase some subtle textures of the background, we threshold the image of gradient-magnitude at 100 (we have normalized the magnitude into the range [0,255]). **Figure 7** shows the effects of such a thresholding. It is possible to locate white aircrafts on gradient thresholding image also. In Section 4, **Table 1** lists the comparative results of two methods. The former method is locating white and colorful aircrafts on gray or gradient thresholding images respectively. The later is locating all aircrafts on gradient thresholding images. The former method has a higher locating accuracy and a higher search efficiency.

Figure 6 shows that even the aircrafts under strong sunshine are segmented clearly in one of the thresholding images. Figure 7 shows that after thresholding at 100, the aircrafts' edge are enhanced and the background textures are reduced.

Algorithm 1 *Main-axis Computing*

Input: a sliding window W_p at position $p = (x_0, y_0)$, ws=window scale, w=1.0×ws, h=1.25×ws.
Output: The main-axis orientation, position, length.

1: **for** $i = 0, 1, \cdots, 39$ **do**
2: Rotate W_p by angle=$i \times 4.5^0$, denote the rotated window as W_{pi}.
3: Compute C_{pi}= the gray projection curve of W_{pi} to horizonal axis, M_{pi}=maximal value of C_{pi}, X_{pi}=x-position of M_{pi}, Y_{pi}=y-position of the geometric center of W_{pi}.
4: **end for**
5: Compute $j = \arg\max_i \{M_{pi} : i = 0, \cdots, 39\}$.
6: Segment R_{pj}= the rectangle region of W_{pj}, which is centered at (X_{pj}, Y_{pj}), width=w, height=h.
7: Compute C_{Rpj}= the gray projection curve of R_{pj} to vertical axis.
8: The main-axis orientation=$j \times 4.5^0$, x-position=X_{pj}, y-position=Y_{pj}, length=width of C_{Rpj}.

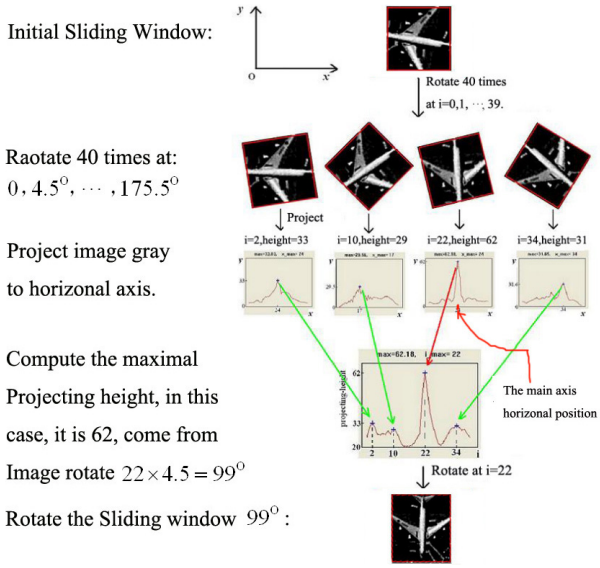

Fig. 8 Only when the window is rotated to the main-axis orientation, can its gray projection curve have the steepest peak.

3.2 Orientation Computing

Computing the orientation, position and length of the main-axis of the object is very important for exact location. The traditional orienting method is based on the minimal geometric 1 or 2 order central moments, some new methods are based on the minimal area of including rectangle [13] or symmetric properties. The geometric central moments are easily disturbed by a small noise, the farer the noise from the central axis, the higher the weights it owned. The method based on area or symmetric property is not stable also, for the reason area and symmetric property are actual geometric property. We proposed a new method based on the maximal projection height. The peek of a projection curve is very stable, because projection is an accumulating procession, its curve peek is rather stable.

Our orientation process is shown in Algorithm 1 and **Fig. 8**.

Figure 9 shows that our method is more robust than other three methods in complex environments.

3.3 Object Locating

On the gray or gradient thresholding images, the sizes of the aircrafts in the airports vary in a wide range, we use multi-scale sliding windows to locate the aircrafts on multiple gray thresholding images or gradient thresholding images. In the experiments of Section 4, three window scales (16, 20, 30) are used. Algorithm 2 and **Fig. 10** show our locating process in details.

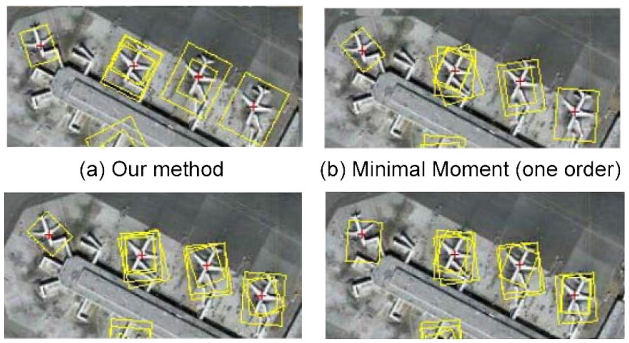

Fig. 9 Partial orientation computing results of four methods. (a) is our method, (b) and (c) are the methods based on minimal one or two order geometric center moments. (d) is the method proposed by Zunic and Kopanja [13], which is based on the minimal including rectangle area.

Algorithm 2 *Object Locating*

Input: an initial sliding window W_p at position $p = (x_0, y_0)$.
Output: The exact location window.

1: Compute the geometric center $p1 = (x_1, y_1)$ of W_p, move the W_p to (x_1, y_1), denote it as W_{p1}.
2: Enlarge the size of W_{p1} twice, compute the new geometric center $p2 = (x_2, y_2)$ of the enlarged window.
3: Move W_{p1} to (x_2, y_2), denote it as W_{p2}.
4: Compute the main-axis of W_{p2}, rotate and move W_{p2} to its main-axis orientation and position, change the window scale to the main-axis length.

At last, some repetitive windows are filtered by a small distance limit (5 pixel). After filtering, all windows are normalized into 48×48 size. Their gray scales are normalized into [0, 255]. Then we sent them to the DNN classifier for feature extracting and aircraft detection. An window is regarded to be a positive sample, if it covers the center of an aircraft, and its scale and orientation are in reasonable ranges that compared with the scale and orientation of the contained aircraft (the allowed scale range is [0.5, 1.5], the orientation range is $[-30^0, +30^0]$).

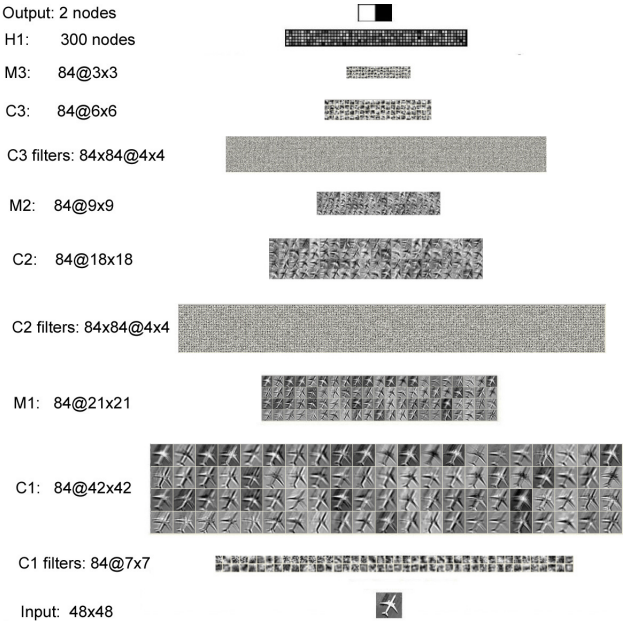

Fig. 11 Structure and parameters of the DNN we used.

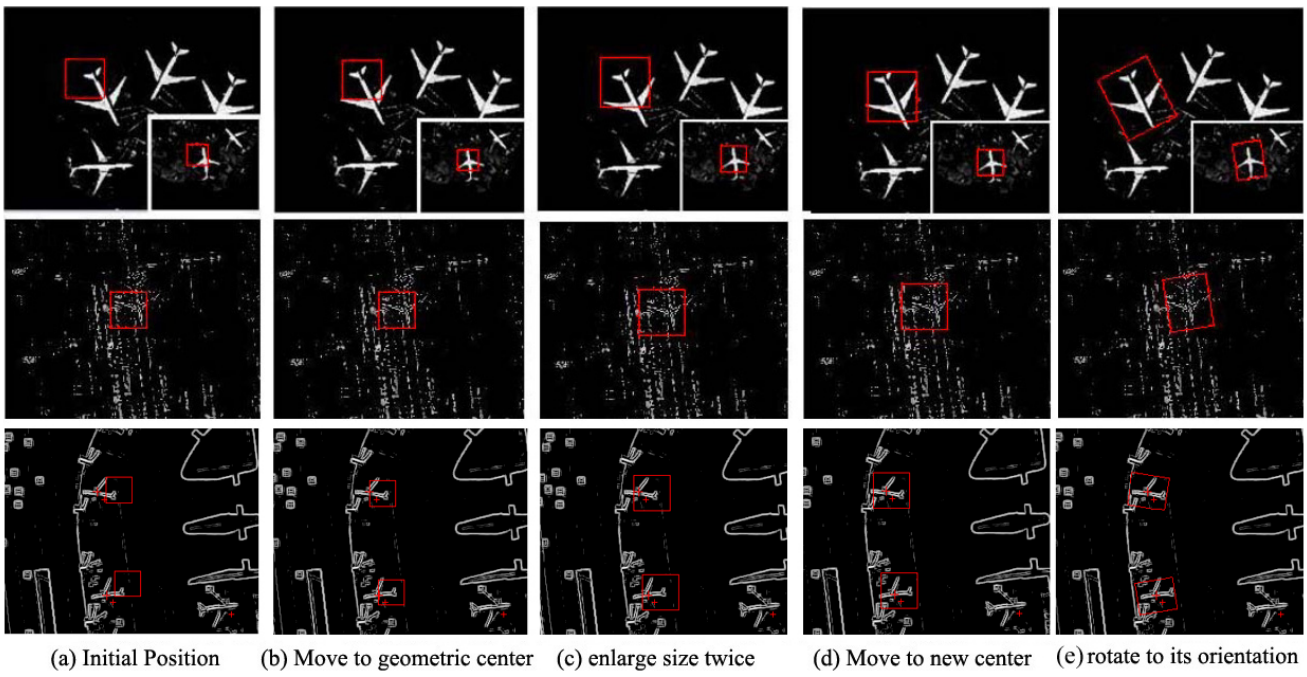

Fig. 10 The four steps of our multi-scales object locating process. The first row is a clear gray thresholding image (threshold=210), the second row is a noisy thresholding image (threshold=250), and the third row is a gradient thresholding image (threshold=100).

3.4 Training DNN

Figure 11 shows the structure of the 9-layer DNN we used in aircraft detection. Here n_l=3, n_m=84. There is only one hidden layer H_1 which has 300 nodes, the output layer has 2 nodes.

The convolutional filter size of C^1 is 7×7, that of C^2 is 4×4, that of C^3 is 4×4. The max-pooling field sizes of M^1, M^2 and M^2 are 2×2. The total node number of M3 is 84×(3×3)=756. This is also the total dimension of the features extracted by the DNN.

The output label of the positive sample is [1,−1], and that of the negative sample is [−1,1]. We trained DNN by the back propagation algorithm on the GPU card, initial weights were set by an uniform random distribution in the range [−0.05, 0.05], all initial biases were set to zero. LearnRate=0.001, Momentum=0, WeightDecay=0, batch size=50. Training is ended when the validation error is near-zero, it usually needs 4-5 days on our GPU cards. After training, we tested all samples in the test set. Test an image in GPU needs about 7 seconds.

4. Experiment

Our database contains 51 airport images (1300 × 950) which were collected from the *Google Earth*. The airports include many famous international city such as Beijing, Los Angeles, Atlanta, Moscow, etc. We selected 26 images, 654 aircrafts as the training set. Other 25 images, 630 aircrafts are used as the test set. The database is very challenging, because some aircrafts are very blurred and their backgrounds are complex.

We define an aircraft is located accurately if it has at least one positive sample.

Table 1 gives the locating accuracy of Algorithm 2 for all 1284 aircrafts in all 51 images. Its first row gives the results of the method that locates white aircrafts on three gray thresholding (thresholds=210, 240, 255), and colorful aircrafts on gradient thresholding (threshold=100) images. The second row shows the results of the method that locates all aircrafts on gradient thresholding (threshold=100) images. It shows the first row has a higher search efficiency and a higher locating accuracy. Where #samples denotes the sample number, we used the samples produced by the first row for the following experiments.

The baseline sliding window method needs about $(\frac{1300\times950}{15\times15} + \frac{1300\times950}{10\times10} + \frac{1300\times950}{8\times8}) \times 51 = 1,893,923$ samples at all. The search efficiency of our method is 12 times (1893923/149923=12.63) more than the baseline sliding window method.

We denote False Alarm Rate (FAR), Precision Rate (PR) and Recall Rate (RR) as:

$$\begin{cases} \text{FAR} = \dfrac{\text{number of false alarms}}{\text{number of aircrafts}} \times 100\% \\ \text{PR} = \dfrac{\text{number of detected aircrafts}}{\text{number of detected objects}} \times 100\% \\ \text{RR} = \dfrac{\text{number of detected aircrafts}}{\text{number of aircrafts}} \times 100\% \end{cases} \quad (7)$$

To be fair and objective, some overlapped false alarms are fused into one alarm.

Table 2 lists the results of five different methods on our aircraft test set, where the input is only a gray image. Here DNN (9-layer) has the structure and parameters as Fig. 11. DBN (800, 800,400,200,2) means that the DBN has 800, 800, 400 and 200 nodes in the first, second, third and fourth hidden layers respectively, and two nodes in the output layer. The HOG feature is computed as [15], its orientation bins is 9. The 48×48 gray image is divided into 1×1+2×2+3×3+4×4+5×5=55 blocks. The HOG dimension is 55×9=495. LBP(8,2) feature means P=8, R=2. LBP(8,3) means P=8, R=3. They include 58 uniform patterns and 1 nonuniform pattern (refer to [32]). The LBP dimension is 59×55=3245. We utilized the rbf kernel, 3000 support vectors in SVM. The kernel parameter is optimized in a range [1/dimension, 30/dimension]. Table 2 reveals that DNN performs better than DBN, and DBN exceeds the traditional methods far away. HOG is better than LBP(8,2), and LBP(8,2) is better than LBP(8,3).

In **Table 3**, G1 is the gray image of the sample. G2 includes the gray image and gradient image of the sample. G4 includes the gray image, a gray thresholding image at 180, a gray thresholding image at 210 and the gradient image of the sample. For example, when input Data is G4, each image of G4 is fed to 21 maps of the C1 layer. This means that the 84 maps of C1 are divided into

Table 2 False alarm rates (%) on test set.

	Recall Rate (%)				
Method	90	85	80	75	70
DNN (9-layer)	47.5	29.3	18.7	12.8	8.80
DBN (800,800,400,200,2)	54.9	35.9	23.7	16.4	10.0
HOG+SVM	101	57.1	38.8	25.6	17.7
LBP(8,2)+SVM	166	118	86.8	65.4	49.4
LBP(8,3)+SVM	192	138	104	82.0	60.4

Table 3 False alarm rates of DNN (9-layer).

	Recall Rate				
Input Data	90	85	80	75	70
G1	47.5	29.3	18.7	12.8	8.80
G2	39.9	24.9	16.2	10.8	7.05
G4	35.1	19.4	12.4	8.28	5.76

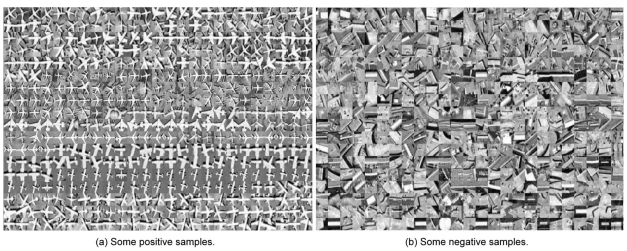

Fig. 12 Partial positive and negative samples in the training database.

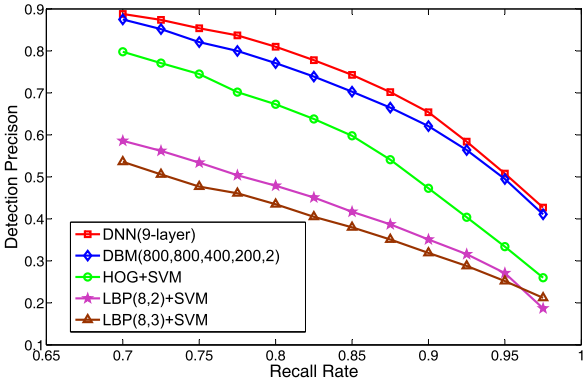

Fig. 13 The RPC curves of five methods in our test database.

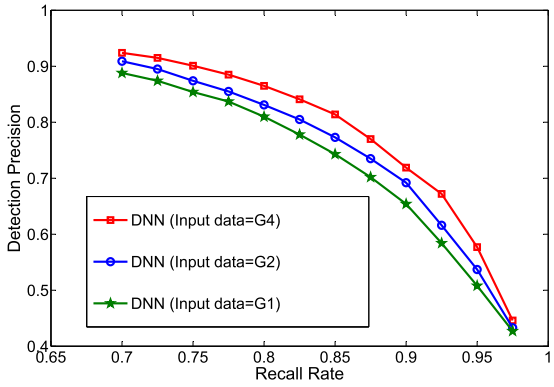

Fig. 14 The RPC curves of the 9-layer DNNs with different input data.

Fig. 15 Partial detection results of DNN in our test database.

four parts, each part has 21 maps and accepts one image of G4, four parts accept the four images of G4 respectively. All samples are preprocessed and fed to C1 in the same multi-images ways, no matter whether it belongs to the train set or the test set.

Table 3 shows that Inputting multi-preprocessed images helps to improve the performance of DNN obviously.

Figure 15 displays partial detection results on the test airport images, owing to the multi-scales object localization method and the powerful DNN detector, most aircrafts are detected repetitively, including some tiny and very blurred aircrafts.

5. Conclusion

Aircraft detection is a difficult problem. We proposed an object location method based on constant multiple gray or gradient thresholding images, which is suitable for white and colorful aircrafts. Our method has a high location precision, with search efficiency 12 times more than the baseline sliding window approach. We trained the Deep convolutional Neural Networks (DNN) as the final detector. Experiments shows that our DNN outperforms another deep machine learning method, the famous Deep Belief Nets (DBN), and DBN outperforms the traditional Feature+Classifier methods with ease. Furthermore, inputting multiple preprocessed images helps to improve the performance of DNN obviously.

Acknowledgments This work was supported in part by the National Natural Science Foundation of China under Grants 91338202, the National Basic Research Program of China (973 Program) Grant 2012CB316300 and the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant XDA06030300).

References

[1] Hsieh, J.-W., Chen, J.-M., Chuang, C.-H. and Fan, K.-C.: Aircraft type recognition in satellite images, *IEE Proceedings Vision, Image and Signal Processing*, Vol.152, No.3, pp.307–315 (2005).
[2] Yildiz, C. and Polat, E.: Detection of stationary aircrafts from satelitte images, *2011 IEEE 19th Conference on Signal Processing and Communications Applications*, pp.515–521 (2011).
[3] Liu, G., Sun, X., Fu, K. and Wang, H.: Aircraft Recognition in High-Resolution Satellite Images Using Coarse-to-Fine Shape Prior, *IEEE Geoscience and Remote Sensing Letters*, Vol.10, No.3, pp.573–577 (2013).
[4] Cai, K., Shao, W., Yin, X. and Liu, G.: Co-Segmentation of Aircrafts from High-resolution Satellite Images, *Proc. ICSP 2012*, pp.993–996 (2012).
[5] Sun, H., Sun, X., Wang, H., Li, Y. and Li, X.: Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model, *IEEE Geoscience and Remote Sensing Letters*, Vol.9, No.1, pp.109–113 (2012).
[6] Li, W., Xiang, S., Wang, H. and Pan, C.: Robust airplane detection in satellite images, *Proc. ICIP*, pp.2877–2880 (2011).
[7] Filippidis, A., Jain, L.C. and Martin, N.: Fusion of intelligent agents for the detection of aircraft in sar images, *IEEE Trans. PAMI*, Vol.22, pp.378–384 (2000).
[8] Tien, S.C., Chia, T.L. and Lu, Y.: Using cross-ratios to model curve data for aircraft recognition, *Pattern Recognit. Lett.*, Vol.24, No.12, pp.2047–2060 (2003).
[9] Xu, C.F. and Duan, H.B.: Artificial bee colony (ABC) optimized edge potential function (EPF) approach to target recognition for low-altitude aircraft, *Pattern Recognit. Lett.*, Vol.31, No.13, pp.1759–1772 (2010).
[10] Scott, G.J., Klaric, M.N., Davis, C.H. and Shyu, C.-R.: Entropy-balanced bitmap tree for shape-based object retrieval from large-scale satellite imagery databases, *IEEE Trans. Geosci. Remote Sens.*, Vol.49, No.5, pp.1603–1616 (2011).
[11] Kembhavi, A., Harwood, D. and Davis, L.S.: Vehicle detection using partial least squares, *IEEE Trans. PAMI*, Vol.63, No.3, pp.1250–1265 (2011).
[12] Ali, K., Fleuret, F., Hasler, D. and Fua, P.: A Real-Time Deformable Detector, *IEEE Trans. PAMI*, Vol.34, No.2, pp.225–239 (2012).
[13] Zunic, J. and Kopanja, L.: On the orientability of shapes, *IEEE Trans. on Image Processing*, Vol.15, No.11, pp.3478–3487 (2006).
[14] Grabner, H., Nguyen, T., Gruber, B. and Bischof, H.: On-Line Boosting-Based Car Detection from Aerial Images, *ISPRS J. Photogrammetry and Remote Sensing*, Vol.63, No.3, pp.382–396 (2008).
[15] Dalal, N. and Triggs, B.: Histograms of oriented gradients for human detection, *Proc. CVPR*, Vol.1, pp.888–893 (2005).
[16] Wiesel, D.H. and Hubel, T.N.: Receptive fields of single neurones in the cats striate cortex, *J. Physiology*, Vol.148, pp.574–591 (1959).
[17] Fukushima, K.: Neocognitron: A self-organizing neural network for a mechanism of pattern recognition unaffected by shift in position, *Biological Cybernetics*, Vol.36, No.4, pp.193–202 (1980).
[18] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P.: Gradient-based learning applied to document recognition, *Proc. IEEE*, Vol.86, No.11, pp.2278–2324 (1998).
[19] Garcia, C. and Delakis, M.: Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.26, No.11, pp.1408–1423 (2004).

[20] Ciresan, D.C., Meier, U. and Schmidhuber, J.: Multi-column Deep Neural Networks for Image Classification, *Proc. Computer Vision and Pattern Recognition*, pp.3642–3649 (June 2012).
[21] Hinton, G.E. and Osindero, S.: A fast learning algorithm for deep belief nets, Neural Computation, Vol.18, pp.1527–1554 (2006).
[22] Hinton, G.E.: A practical guide to training restricted boltzmann machines, available from ⟨http://www.csri.utoronto.ca/hin-ton/absps/⟩, pp.1–20 (Aug. 2010).
[23] Hinton, G.E.: Reducing the dimensionality of data with neural networks, *Science*, Vol.313, pp.504–507 (2006).
[24] Yu, D., Deng, L. and Dahl, G.: Roles of pretraining and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition, *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning* (2010).
[25] Mohamed, A., Yu, D. and Deng, L.: Investigation of full-sequence training of deep belief networks for speech recognition, *Proc. Interspeech 2010*, pp.1692–1695 (2010).
[26] Sarikaya, R. and Hinton, G.E.: Deep belief nets for natural language callcrouting, *Proc. ICASSP*, pp.5680–5683 (2011).
[27] Seide, F., Li, G. and Yu, D.: Conversational speech transcription using context-dependent deep neural networks, *Proc. Interspeech 2011* (2011).
[28] Mohamed, A., Dahl, G.E. and Hinton, G.: Acoustic Modeling Using Deep Belief Networks, *IEEE Trans. Audio, Speech, and Language Processing*, Vol.20, No.1, pp.14–22 (2012).
[29] Dahl, G., Yu, D., Deng, L. and Acero, A.: Context-dependent pre-trained deep neural networks for large vocabulary speech recognition, *IEEE Trans. Speech and Audio Processing*, Vol.20, No.1, pp.30–42 (2012).
[30] Yu, D., Seide, F., Li, G., Li, J. and Seltzer, M.: Why deep neural networks are promising for large vocabulary speech recognition, *IEEE Trans. Audio, Speech, and Language Processing* (2012).
[31] Daugman, J.G.: Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression, *IEEE Trans. Acoust., Speech, Signal Processing*, Vol.36, No.7, pp.1169–1179 (1988).
[32] Ojala, T., Pietikainen, M. and Maenpaa, T.: Multiresolution Gray Scale and Rotation Invariant Texture Classification with Local Binary Patterns, *IEEE Trans. PAMI*, Vol.24, No.7, pp.971–987 (2002).
[33] Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints, *International Journal of Computer Vision*, Vol.60, No.2, pp.91–110 (2004).

Xueyun Chen received his B.S. degree in Aircraft structure strength from the University of National Defence technology, China, in 1990, M.S. degree in industrial automation from Huazhong Science and Technology University, China, in 1997, and Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 2014. He is currently an associate professor in the Electrical Engineering College, Guangxi University. His research interests include deep convolutional neural networks, super neural networks.

Shiming Xiang received his B.S. degree in mathematics from Chongqing Normal University, China, in 1993, M.S. degree from Chongqing University, China, in 1996, and Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2004. From 1996 to 2001, he was a lecturer with the Huazhong University of Science and Technology, Wuhan, China. He was a postdoctoral researcher with the Department of Automation, Tsinghua University, Beijing, China, until 2006. He is currently an associate professor with the Institute of Automation, Chinese Academy of Sciences, Beijing. His research interests include pattern recognition and image processing. He is a member of IEEE.

Cheng-Lin Liu received his B.S. degree in electronic engineering from Wuhan University, China, M.E. degree in electronic engineering from Beijing Polytechnic University, China, and Ph.D. degree in pattern recognition and intelligent control from the Chinese Academy of Sciences, Beijing, China, in 1989, 1992, and 1995, respectively. He is a professor at the National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences, Beijing, China, and is now the deputy director of the laboratory. He was a postdoctoral fellow at Korea Advanced Institute of Science and Technology and later at Tokyo University of Agriculture and Technology from March 1996 to March 1999. From 1999 to 2004, he was a research staff member and later a senior researcher at the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan. His research interests include pattern recognition, image processing, neural networks, machine learning, and especially the applications to character recognition and document analysis. He has published more than 170 technical papers at prestigious international journals and conferences. He is on the editorial board of journals Pattern Recognition, Image and Vision Computing, and International Journal on Document Analysis and Recognition. He is a fellow of IAPR and IEEE.

Chun-Hong Pan received his B.S. degree in automatic control from Tsinghua University, Beijing, China, in 1987, M.S. degree from the Shanghai Institute of Optics and Fine Mechanics, Chinese Academy of Sciences, China, in 1990, and Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 2000. He is currently a professor in the National Laboratory of Pattern Recognition of Institute of Automation, Chinese Academy of Sciences. His research interests include computer vision, image processing, and remote sensing.

(Communicated by *Yi-Ping Hung*)