The Journal of Engineering

The 2nd 2018 Asian Conference on Artificial Intelligence Technology (ACAIT 2018)

IET Journals
The Institution of Engineering and Technology

Combining HWEBING and HOG-MLBP features for pedestrian detection

eISSN 2051-3305
Received on 18th July 2018
Accepted on 26th July 2018
E-First on 25th September 2018
doi: 10.1049/joe.2018.8308
www.ietdl.org

Ling Xiao[1,2], Yongjun Zhang[1,2] ✉, Jian Zhang[3], Qian Wang[1,2], Yuewei Li[1,2]

[1]*Key Laboratory of Intelligent Medical Image Analysis and Precise Diagnosis of Guizhou Province, Guiyang, People's Republic of China*
[2]*School of Computer Science and Technology, Guizhou University, Guiyang, People's Republic of China*
[3]*School of Computing and Communications, University of Technology Sydney, Sydney, Australia*
✉ *E-mail: zyj6667@126.com*

Abstract: Pedestrian detection has vital value in many areas such as driver assistance systems, driverless cars, intelligent tourism systems etc., but there are some difficulties that need to be solved. The algorithm with high detection rate is complex and requires substantial time. Therefore, how to improve the detection accuracy and speed has become the key of pedestrian detection. For these reasons, firstly, an improved algorithm, called hash and window enhancement of binarised normed gradients (HWEBING), based on binarised normed gradients feature is proposed. Subsequently, the authors present an improved local texture feature, namely mean of local binary pattern (MLBP), based on uniform pattern local binary pattern (ULBP) for increasing the detection rate. Finally, after using the HWEBING algorithm to get the candidate windows, the combination of MLBP feature and histograms of oriented gradients feature is extracted from these windows to further enhance the detection accuracy. Experimental results reveal that speed of using the HWEBING algorithm for pre-detection is 5.5 times faster than the traditional method of pedestrian detection. Furthermore, the detection rate of MLBP feature is 3.5 and 2.1% higher than those of ULBP and basic pattern local binary pattern (Basic-LBP), respectively.

1 Introduction

Pedestrian detection is a very important technology in computer vision and pattern recognition. Its purpose is to accurately locate pedestrians in images or video sequences, yet practical applications of pedestrian detection will have quantities of difficulties in that both driver assistance systems and unmanned cars require algorithms to meet high detection accuracy and real-time. Consequently, it is of great value to design an ideal algorithm with real-time response. There are mainly two methods for pedestrian detection. One is based on background modelling [1] and the other is based on statistical learning [2].The mainstream method is based on the statistical learning method which has better accuracy and robustness. Its basic standpoint is to extract the feature of samples, then use the classifier to train models, and finally detect images or video sequences by the model. Frequently used features of pedestrian detection are historgram of oriented gradients (HOG), local binary pattern (LBP), Haar-like, colour self-similarities (CSS), integral channel features (ICF) etc. The HOG feature proposed by Dalal and Triggs [3] which described pedestrians' edge information well had made breakthrough progress by combining with the support vector machine (SVM) classifier [4]. Maji *et al.* [5] combined the HOG feature with the HIKSVM classifier [6] to greatly improve the detection rate, yet the HIKSVM classifier had high computational complexity. The LBP feature [7] expressing texture information of the image had the property of greyscale and rotation invariances, so it had wide applications in many fields. The local ternary pattern [8] feature proposed by Tan and Triggs had preferable anti-noise effect, however, reducing the invariance of local light. Mu *et al.* [9] presented the semantic LBP and Fourier LBP descriptors that were propitious to describe texture information of pedestrians. Heikkil *et al.* [10] proposed the centre symmetric-LBP that was more commonly used in image matching. Papageorgiou and Poggio [11] proposed the combination of Haar-like feature with polynomial kernel SVM classifier. Nevertheless, representation capability of the Haar-like feature was poor, which resulted in higher false detection rate. The CSS feature [12] based on the colour information divided the image into multiple blocks, then counted the colour histogram of each block and calculated the similarity between histograms. CSS feature was easily affected by factors like lighting, but it can be complemented with HOG and LBP features to achieve effective detection results. The ICF feature [13] combined various features from different angles, which achieved good detection results, yet the detection speed was slow. The combination of multiple features can achieve better results in pedestrian detection. Therefore, Wang [14] combined LBP feature with HOG feature to deal with partial occlusion of human body. Walk *et al.* [12] also obtained better results by using the integration of CSS feature and HOG feature.

The above methods are based on the traditional pyramid scanning method to detect images. The number of sliding windows for each image is directly proportional to sizes, which leads to slow speed of detecting the large image. Moreover, the majority of windows are non-existent pedestrian, so reducing the number of windows can greatly improve speed. The fastest pedestrian detector in the West (FPDW) method proposed by Dollár *et al.* [15] used single-scale features to evaluate the features of adjacent scales to speed, but only for limited features. Benenson *et al.* [16] estimated different-scale models by training single-scale models, which improved speed, but only for special scenes. Besides, these methods require GPU to accelerate. Cheng *et al.* [17] proposed the BING feature that had very fast speed and simple calculation, but the detection rate of BING applied in pedestrian detection is relatively low, so it still needs to be improved.

In recent years, deep learning has been widely used in pedestrian detection, like Faster R-CNN [18], YOLO [19] etc. However, the method based on deep learning has high training cost, complicated verification and poor generalisation ability of models.

From the perspective of improving the detection rate and speed, first of all, this paper proposes an improved HWEBING algorithm based on BING feature to improve speed. The HWEBING algorithm firstly optimises samples by the K-Means++ algorithm in training phase. Then using the hash table to preserve the convolution result that may be calculated in the detection stage improves speed of detection. Additionally, the window enhancement method is proposed for the HWEBING algorithm because complicated objects cannot get the correct window score. After filtering out the windows that may not be objects for each

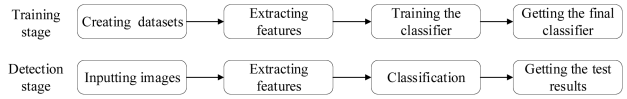

Fig. 1 *Traditional pedestrian detection framework*

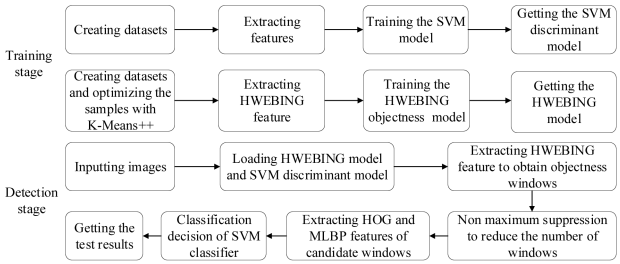

Fig. 2 *Pedestrian detection framework in this paper*

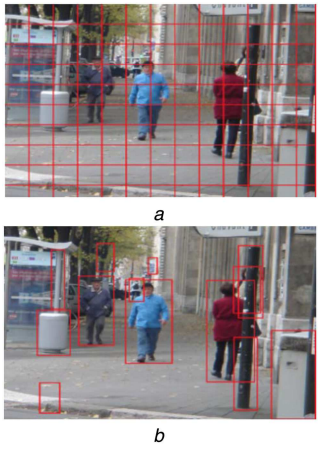

Fig. 3 *Comparison between the traditional sliding scanning window method and the object detection method*
(a) The traditional sliding scanning window method, ***(b)*** The object detection method

image, an improved local texture feature called MLBP is proposed in the process of extracting features of objectness windows to increase the detection rate. Later, the MLBP feature is combined with the HOG feature to accurately detect the candidate windows obtained by the HWEBING algorithm to further enhance the detection rate. Experimental results show that using HWEBING algorithm combined with the HOG-MLBP feature can effectively improve the detection rate and speed of pedestrian detection.

2 Comparison of frameworks for pedestrian detection

In the training stage, the traditional pedestrian detection framework obtains pedestrian detectors by extracting features from datasets and training samples with classifiers. In the detection stage, images to be detected are taken as inputs of the detectors, which can get test results. The specific process is shown in Fig. 1.

The pedestrian detection framework of this paper is shown in Fig. 2. Above all, the K-Means++ algorithm clusters the samples in the training stage. Afterwards HWEBING feature is extracted and trained to obtain the objectness model. In the detection stage, using the HWEBING feature for pre-detection obtains windows that may be objects in each image, and then windows with high scores are selected as candidate windows handled by non-maximum suppression (NMS). Eventually, the HOG and MLBP features are extracted on final windows.

3 BING feature and HWEBING algorithm

3.1 BING feature

For the image with sizes of $M \times M$, the conventional sliding scanning window method may generate M^4 candidate windows, as shown in Fig. 3*a*. For large images, the number of candidate windows greatly limits speed of detection and therefore reducing the number of windows will undoubtedly improve the speed. If there is the transcendental knowledge that can pick out windows of possible objects in each image, the speed will be raised.

The BING feature proposed by Cheng *et al.* is based on the idea of object detection. It reduces the scale of windows to uniform size (8×8) and the gradient magnitude graph of the scaled image is obtained. The 64 dimensions of normalised gradients are defined as the normed gradients (NG) feature of the window. Gradients are important information for distinguishing objects and backgrounds for the gradient magnitude graph of objects in closed contours has relatively high similarity. The pre-detection of the BING feature drastically reduces the number of windows, as shown in Fig. 3*b*.

The BING feature obtains scores of windows through machine learning, the score formula as shown in

$$s_l = \langle w, g_l \rangle \tag{1}$$

s_l is the score of the window and w is the SVM model obtained by training the 64-dimensional gradient feature. g_l is the NG feature of the window and l is the size and position of the window.

The NMS method can obtain the recommended windows for each scale and maximum amounts of windows for every scale in this paper are 130. Usually, some windows may not contain objects. For instance, the aspect ratio of human body mainly between 1:4 and 1:1 conforms to normal distribution. Therefore, in order to allow the window to have a more correct score, two impact factors v_i and t_i are added. The final score formula is shown in (2) where v_i and t_i are different learning coefficients of each scale i:

$$o_l = v_i \cdot s_l + t_i \tag{2}$$

The BING feature is based on the NG feature for the binarisation increases the speed of detection. The 64-dimensional model w is approximately regarded as the combination of basis vectors, as shown in (3) where N_w is the number of basis vectors. a_j ($a_j \varepsilon \{-1,1\}^{64}$) is the basis vector and β_j is the coefficient of a_j. Furthermore, a_j can be expressed as a two-dimensional vector and its complement, as shown in (4) where $a_j^+ \varepsilon \{0,1\}^{64}$. Finally, the binarisation model can use bitwise operation to speed, the formula shown in (5) where b is the binarised feature:

$$w \simeq \sum_{j=1}^{N_w} \beta_j a_j \tag{3}$$

$$a_j = a_j^+ - \overline{a_j^+} \tag{4}$$

$$\langle w, b \rangle \simeq \sum_{j=1}^{N_w} \beta_j \left(2\langle a_j^+, b \rangle - |b|\right) \tag{5}$$

In order to accelerate the calculation, the NG feature is also approximated and N_g bits of each feature replace the NG feature. Therefore, the formula of the NG feature is shown in (6) where $b_{k,l}$ is the BING feature:

$$g_l = \sum_{k=1}^{N_g} 2^{8-k} b_{k,l} \tag{6}$$

3.2 HWEBING algorithm

The convolution of BING feature is performed on each pixel of the gradient image with the trained BING template. When the 8×8 template convolves for each pixel, 64 multiplications and 63 additions are required, which is a very time-consuming operation for the detection process.

The HWEBING algorithm makes the following improvements on the BING feature. Firstly, the K-Means++ algorithm optimises the training samples. Secondly, for time-consuming convolution in the BING feature, the HWEBING algorithm establishes a hash

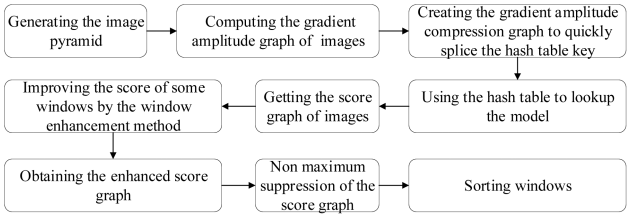

Fig. 4 *Flowchart of extracting HWEBING feature*

Fig. 5 *Principle of window enhancement*

table during the training phase and results of the convolution are stored in the hash table ahead of time. When pixels are convolved in the detection stage, seeking the hash table directly gets results of the convolution. Finally, when scores of objectness windows are calculated, the scores of complex objects are enhanced. The flowchart of extracting HWEBING feature is shown in Fig. 4.

3.2.1 The optimisation of samples by K-Means++ algorithm: K-Means++ is an improvement on the K-Means [20], which is the common clustering method. The K-Means algorithm randomly selects some initial points as cluster centres, then calculates the distance between each sample point and the cluster centre and classifies it as the closest category to central points. After the samples are initially clustered, the centre point of each category is recalculated according to the above method until the centre point of the class is stable, and this point is taken as the centroid of the category. The K-Means algorithm needs to be iterated many times, and thus the convergence speed is slow. Moreover, randomly selecting the initial point has great influences on the clustering effect and the convergence speed. However, K-Means++ makes some improvements on K-Means for the selection of initial points. First, the samples are divided into K classes, and then random data sets are generated. In these data sets, one sample point is arbitrarily chosen as the centre point of the first class. Afterwards selecting the next cluster centre according to probability, the probability formula is shown in (7) where D_i is the square of distance between the ith sample point and its nearest centre, and sum(D_i) is the sum of all the nearest distances. Finally, this operation is repeated until the K centre points are found, and the K-Means algorithm is performed after finding the K centre points:

$$P(i) = \frac{Di}{\text{sum}(Di)} \tag{7}$$

3.2.2 Method of hash and window enhancement: Hash lookup mainly includes two steps: building the hash table in the training phase and looking for the hash table in the detection phase. Hash tables are established for the first four columns and the last four columns of each row for the 8×8 model trained by the first-level SVM, so each model corresponds with 16 hash tables. In order to speed up the calculation of the gradient amplitude, the first four bits are used for each gradient magnitude. The process of creating the hash table is as follows, where $h(x,y)$ is a hash table and x ($x\varepsilon\{0,1\}$) is the first four or the last four. y ($y\varepsilon(0,7)$) is the number of rows and $w(x,y)$ is the model value at (x,y):

(i) Assuming $i=0$, $b1=i\&0xf$, $b2=(i>>4)\&0xf$, $b3=(i>>8)\&0xf$, $b4=(i>>12)\&0xf$.

(ii) For $y\varepsilon(0,7)$, we calculate the value of $h(x,y)$ shown in (8):

$$\begin{aligned} h(0,y) &= b1^{*}w(y,3) + b2^{*}w(y,2) + b3^{*}w(y,1) + b4^{*}w(y,0) \\ h(1,y) &= b1^{*}w(y,7) + b2^{*}w(y,6) + b3^{*}w(y,5) + b4^{*}w(y,4) \end{aligned} \tag{8}$$

(iii) $i=i+1$, the above process is repeated until $i \geq 2^{16}$.

After establishing the hash table, the lookup table can obtain results of the convolution; but when looking for the hash table, we need to create a gradient amplitude compression graph to quickly join the table key. The value at (x,y) in this matrix is the first four of the last eight values at the original gradient magnitude graph (x,y).

The BING feature cannot give completely correct scores when calculating the window scores for complex objects. As shown in Fig. 5, the jacket and trouser on the human body may be viewed as two different objects by the BING feature, resulting in higher scores for the proposed windows of the two parts. Furthermore, the gradients of the join between the two parts are too different. It is not always possible to form a good closed contour for the pedestrian, which may cause the score of the blue window to decrease.

Therefore, this paper proposes the window enhancement method. For a window with sizes of $2^m \times 2^n$, it is enhanced by using the score of the window that is half smaller than it, the formula shown in (9):

$$\begin{aligned} S'_{2^m \times 2^n}(x,y) = S_{2^m \times 2^n}(x,y) + A \times S_{2^{m-1} \times 2^n}(x+i,y) \\ + B \times S_{2^m \times 2^{n-1}}(x,y+i) \end{aligned} \tag{9}$$

$S_{2^m \times 2^n}(x,\ y)$ is the score of window sizes of $2^m \times 2^n$ in the filtered score graph at (x,y). $S'_{2^m \times 2^n}(x,y)$ is the enhancement score, and A and B are the coefficients for enhancement at $(x+i,\ y)$ and $(x,\ y+i)$, respectively. Massive experiments prove that when $i=3$, A and B are equal to 0.49, the effect of window enhancement is the most remarkable and stable.

3.2.3 NMS method: The overlapped area of adjacent two windows obtained by using the sliding scanning window method or the object detection method is large, so NMS of the score graph is required. There are two methods for NMS:

(i) We sort by the window scores from largest to smallest and then select the point with the highest score as the suggestion window. Finally, we calculate the overlap ratio between the other windows and the window with the highest score. When the overlap ratio reaches certain threshold, the window will be removed. This process is repeated until all windows are calculated.
(ii) Each value in the filtered score graph is processed by the mean filter. When the value after filtering reaches certain threshold, which indicates that this point may be a maximum point, it is retained, and otherwise it is removed. Then these values are arrayed from the largest to the smallest, and a threshold t is set to remove other windows that are not in the $t \times t$ regions.

The first NMS method requires iterative sorts, and thus the time complexity is higher, but the obtained windows are more accurate. The second method only needs to be sorted once, which is more suitable for rapid pedestrian detection. Therefore, the NMS method used in this paper is the first method.

4 HOG-MLBP feature

4.1 MLBP feature

The LBP feature that describes image texture information has simple calculation and is insensitive to illumination changes. It takes the greyscale value of central point for each block as the threshold value. Then calculating the difference between the greyscale value of neighbourhood pixels for blocks and the threshold value gets the binary sequence. By converting the binary sequence into decimal can obtain the LBP value of the pixel point. LBP mainly includes Basic-LBP, ULBP and rotation invariant

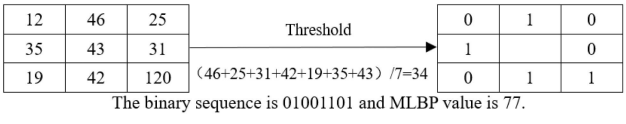

Fig. 6 *Example of MLBP feature calculation*

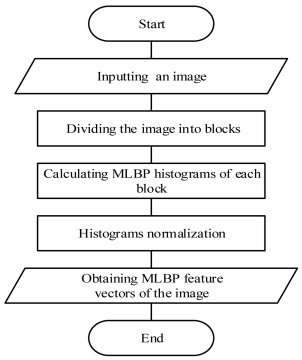

Fig. 7 *Flowchart of extracting MLBP feature*

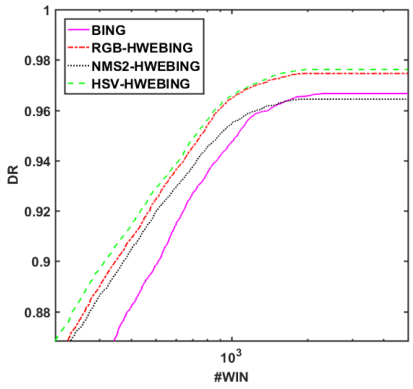

Fig. 8 *Comparison of BING and HWEBING algorithms*

pattern LBP. Too high dimension of Basic-LBP leads to slow detection speed. The ULBP is an improvement on the Basic-LBP, reducing the dimension from 26,880 to 6195, which greatly improves speed. The rotation invariant pattern LBP is an improvement on the ULBP, but the frequency of feature vectors will change and the angle of neighbourhood pixels only has a rough segmentation, so the differentiability of the rotation invariant pattern LBP is not strong enough.

The ULBP feature takes greyscale value of the central pixel on each region as a threshold, which does not take into account the abrupt change between the centre pixel and the surrounding pixels and lacks smooth stability.

Therefore, the MLBP feature makes improvements on this defect. This paper extracts the square MLBP feature (square represents neighbourhood pixels in a square area) and circular MLBP feature (circle represents neighbourhood pixels in a circular area). The MLBP feature divides each image into multiple blocks. Firstly, we find the maximum greyscale value and the minimum greyscale value of the neighbourhood pixels in block and calculate the mean value of all pixels in the block except the maximum and minimum values. Then the mean value is taken as the threshold of this block. Finally, we obtain the MLBP value of each point in the block according to the calculating method of ULBP feature.

Removing the maximum and minimum values decreases the influence of noise on image. There are some differences between the pedestrians and background in the greyscale value, and thus stability of the threshold can be increased by obtaining the mean value. Besides, the texture information of the local region can be described more comprehensively by adding greyscale value of the central point. Here, taking square block of 3×3 as an example, the calculation of MLBP is shown in Fig. 6.

From Fig. 6 we can clearly know the formula for calculating the threshold of MLBP, as shown in

$$Ln = \frac{1}{p-1}\sum_{i=1}^{p}(Li + Lc - \text{Max} - \text{Min}) \tag{10}$$

In the above formula, p is the number of neighbourhood pixels in the block. Max and Min are the maximum value and the minimum value of neighbourhood points in the block, respectively. The L_i and L_c are the greyscale values of i and the centre point for the block, respectively.

After getting the threshold L_n, the value of MLBP is shown in (11), when $i > 0$, $t(i) = 1$, $i \leq 0$, $t(i) = 0$:

$$\text{MLBP}(x, y) = \sum_{i=1}^{p} t(Li - Ln) \cdot 2^{i-1} \tag{11}$$

The flowchart of extracting MLBP feature is shown in Fig. 7.

4.2 HOG-MLBP feature

The HOG feature describes edge information of the image and the LBP feature describes texture information of the image, which is complementary to certain extent. Therefore, combining HOG feature with MLBP feature can further improve the detection rate. The HOG-MLBP feature is to extract MLBP and HOG features on the image, and then the two feature vectors are cascaded.

5 Analysis of results

The experimental platform is on the PC with Intel i5@3.2 GHz CPU and 8G RAM. The experimental environment is Visual Studio 2013 + OpenCV2.4.10. The pedestrian SVM discriminant model uses the INRIA person dataset which has 2416 positive training samples, 1218 negative training samples, 1126 positive testing samples, 453 negative testing samples and 288 detection samples.

Training BING model and HWEBING model uses the VOC2007 dataset that has 9963 images, including 5011 training and validation samples and 4952 test samples.

Experiments measure the performance of algorithms from the detection rate and time to elaborate. The evaluation standard used in this paper includes the detection rate curve, the FPPW curve and the FPPI curve.

5.1 Result of improved BING feature

The HWEBING feature is extracted from RGB and HSV spaces, respectively, and the result shows that the HWEBING feature has higher detection rate in HSV space, as shown in Fig. 8. The results of two NMS methods mentioned above are shown in Fig. 8. The first method requires multiple sorts, resulting in higher time complexity, but the detection rate is higher. The second method is sorted only once. Considering the detection rate, subsequent extraction of the HWEBING feature uses the first NMS method.

When the number of sampling windows is 1000, the detection rate of BING is 95.8% while the detection rate of HWEBING is 96.95%. The detection rate of HWEBING is 1.15% higher than that of BING.

5.2 Comparison of MLBP with other LBP patterns

The square MLBP feature is compared with the ULBP, the Basic-LBP, the circular MLBP, the circular ULBP and the circular Basic-LBP, as shown in Fig. 9. When FPPW = 10^{-4}, the detection rate of MLBP feature is higher than the ULBP and the Basic-LBP. Moreover, the detection rate of MLBP + HOG is better than ULBP + HOG. Their detection rates are shown in Table 1. However, calculations of circular MLBP are more complex, so the subsequent extraction of MLBP feature is based on square MLBP.

5.3 Combination of BING, HWEBING and HOG, HOG + MLBP

(i) *Comparison of BING + HOG and HWEBING + HOG*: In order to test the performance of BING and HWEBING, the HOG feature

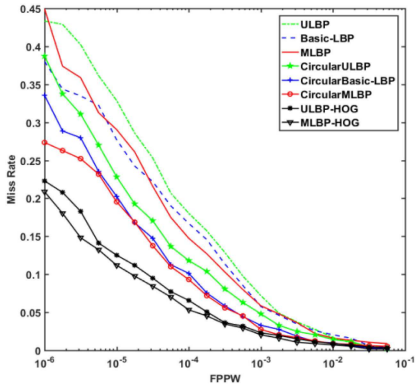

Fig. 9 *Comparison of MLBP feature with other LBP patterns*

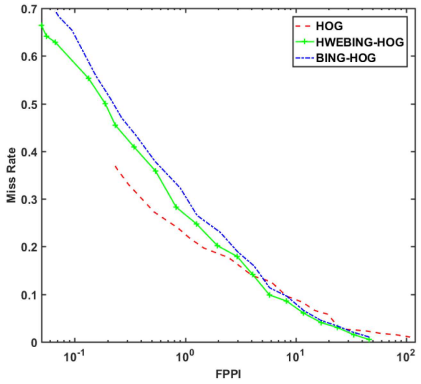

Fig. 10 *Comparison of BING + HOG and HWEBING + HOG*

Table 1 Detection rates for LBP features (FPPW = 10^{-4})

Feature	Dimension	Detection rate, %
circular MLBP	6195	90.8
circular ULBP	6195	88.2
circular Basic-LBP	26880	89.7
square MLBP	6195	85.4
square ULBP	6195	81.9
square Basic-LBP	26880	83.3
square MLBP + HOG	9975	94.5
square ULBP + HOG	9975	93.1

Table 2 Detection rates of algorithms trained by the SVM and HIKSVM classifiers (FPPI = 1)

	Detection algorithm	Detection rate, %
SVM	HOG	76.9
	BING + HOG	69.7
	HWEBING + HOG	72.5
	BING + HOG + MLBP	74.5
	HWEBING + HOG + MLBP	74.9
HIKSVM	BING + HOG + MLBP	74.6
	HWEBING + HOG + MLBP	76.1

is used for accurate detection after the pre-detection by BING and HWEBING. The experimental result is shown in Fig. 10. When FPPI = 1, the detection rate of HOG is higher than BING + HOG and HWEBING + HOG. Meanwhile, the detection rate of HWEBING + HOG is higher than BING + HOG. Their detection rates are shown in Table 2. Since the pre-detection of images will lose some windows that may be the human body to certain extent, the detection rate will decrease; but the pre-detection reduces a large number of candidate windows and can greatly improve the detection speed.

(ii) *Comparison between BING + HOG + MLBP and HWEBING + HOG + MLBP on SVM and HIKSVM classifiers*: After obtaining candidate windows with BING and HWEBING, the HOG and MLBP features are extracted from these windows. The results trained by the SVM and the HIKSVM classifiers are shown in Fig. 11. When FPPI = 1, the detection rate of HWEBING + HOG + MLBP is 0.4% higher than that of BING + HOG + MLBP on the SVM classifier. Moreover, the detection rate of HWEBING + HOG + MLBP is 1.5% higher than that of BING + HOG + MLBP on the HIKSVM classifier. After using the HWEBING algorithm to obtain the candidate windows and extracting the HOG and MLBP features from these windows, the detection rate trained by the HIKSVM classifier is 1.2% higher than the SVM classifier. Their detection rates are shown in Table 2.

(iii) *Comparison of BING + HOG, HWEBING + HOG and BING + HOG + MLBP, HWEBING + HOG + MLBP*: Combining the HOG and MLBP features improves the detection rate of the HOG feature. As shown in Fig. 12, when FPPI = 1, detection rates of BING + HOG + MLBP and HWEBING + HOG + MLBP are 4.8 and 2.4% higher than those of BING + HOG and HWEBING + HOG, respectively, which shows that the combination of HOG and MLBP features can achieve better results.

The pre-detection of HWEBING selects at most 130 windows from each image. The number of windows is much smaller than the traditional pyramid scanning method, and thus it is lower than the traditional method in detection accuracy. However, it can improve the detection speed to a great extent. The hash method of HWEBING improves the speed of BING, and the window enhancement method increases the scores of windows that are more likely to be objects, hence the window has higher confidence level. The enhanced window method of HWEBING will improve the detection rate than the BING feature to some extent. As we can know from Table 2, when FPPI = 1, the detection rate of HWEBING + HOG is 2.8% higher than that of BING + HOG, but the detection rate of HWEBING + HOG + MLBP is 0.4% higher than that of BING + HOG + MLBP for the combination of multiple methods attains the state of overfitting. Although the detection rate of the two methods is almost identical, the speed of the HWEBING + HOG + MLBP is faster than the BING + HOG + MLBP, which is proved by detecting the 288 samples of INRIA in the following section.

(iv) *Detection time of HOG + MLBP, BING + HOG + MLBP and HWEBING + HOG + MLBP*: The speed is also a criterion to judge pedestrian detection algorithm. The following three methods are used to detect the 288 samples of INRIA that have a great many different size pictures, the results are shown in Table 3. After using BING and HWEBING to roughly detect images, a large number of windows can be filtered out, which is very necessary for real-time pedestrian detection. When speed is accelerated, multiple features or complex classifiers can be used to further increase the detection rate. According to the following table, when the HOG feature is combined with the MLBP feature and the HWEBING algorithm is used for pre-detection of images, the speed is improved by 5.5 times compared with the traditional method.

When detecting 288 samples of the INRIA dataset, the detection time of the HWEBING + HOG + MLBP algorithm is 301 s less than the BING + HOG + MLBP algorithm. If the number of samples is larger, the improvement of speed for the HWEBING + HOG + MLBP algorithm is more obvious. Consequently, the HWEBING + HOG + MLBP algorithm can obtain better results by taking into account the detection rate and speed.

5.4 Detection results

Based on the above results, it can be concluded that the MLBP feature is better than the ULBP and the Basic-LBP, and furthermore the HWEBING algorithm is also superior to BING in detection rate and speed. Combining the HWEBING algorithm with the HOG and MLBP features can achieve better detection results. The results of detecting 288 samples by using BING + HOG, HWEBING + HOG, HWEBING + HOG + MLBP algorithms are shown in Fig. 13. The first, second and third columns of each image are the results obtained by the BING + HOG, HWEBING + HOG, and HWEBING + HOG + MLBP algorithms, respectively. It

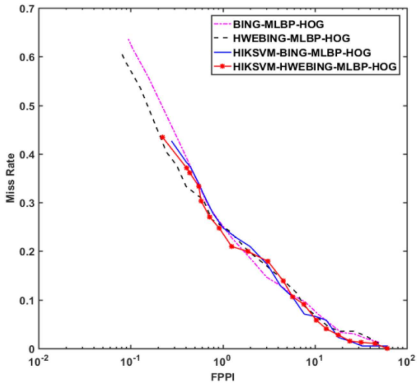

Fig. 11 *Comparison of BING + HOG + MLBP and HWEBING + HOG + MLBP trained by SVM and HIKSVM*

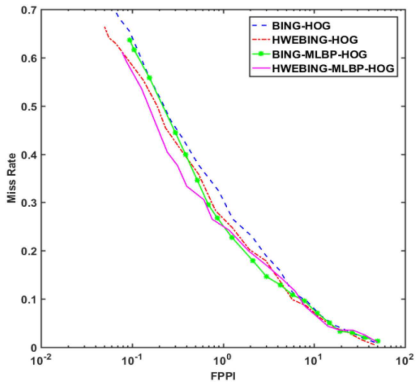

Fig. 12 *Comparison of BING + HOG, HWEBING + HOG and BING + HOG + MLBP, HWEBING + HOG + MLBP*

can be seen from the figure that it is best to use the HWEBING algorithm to obtain the candidate windows and extract the HOG and MLBP features from these windows.

6 Conclusion

Firstly, this paper proposes the improved HWEBING algorithm based on the BING feature for speeding up detection. Multiple experiments have proved that HWEBING is superior to BING in detection rate and speed. After using the HWEBING algorithm to get candidate windows of each image, the improved local texture feature called MLBP is proposed in the process of extracting features. Experimental results show that the detection rate of the MLBP feature is higher than those of the ULBP and the Basic-LBP. Finally, the combination of MLBP and HOG features further improves the detection rate. Therefore, the combination of the HWEBING algorithm and the HOG-MLBP feature proposed in this paper can effectively improve the detection rate and speed of pedestrian detection. However, the NMS method may remove some human windows and the HWEBING algorithm obtains limited windows after pre-detection, so the detection rate of combining the HWEBING algorithm with the HOG-MLBP feature can be improved in the future.

7 Acknowledgments

This work was supported by the Joint Fund of Department of Science and Technology of Guizhou Province and Guizhou University under grant: LH [2014]7635, Research Foundation for Advanced Talents of Guizhou University under grant: (2016) No. 49, Key Supported Disciplines of Guizhou Province - Computer Application Technology (No.QianXueWeiHeZi ZDXX[2016]20), Specialized Fund for Science and Technology Platform and Talent Team Project of Guizhou Province(No.QianKeHePingTaiRenCai [2016]5609), and the work was also supported by National Natural Science Foundation of China (61462013, 61661010).

Table 3 Comparison of detection time

Detection algorithm	Test samples	Total time, s	Average time, s
HOG + MLBP	INRIA 288 samples	13482.3	46.8
BING + HOG + MLBP	INRIA 288 samples	2749.5	9.5
HWEBING + HOG + MLBP	INRIA 288 samples	2448.5	8.5

Fig. 13 *Comparison of detection results for different algorithms*

8 References

[1] Zhong, Z., Zhang, B., Lu, G., *et al.*: 'An adaptive background modeling method for foreground segmentation', *IEEE Trans. Intell. Transp. Syst.*, 2017, **18**, (5), pp. 1109–1121

[2] Yang, T., Jing, L.I., Pan, Q., *et al.*: 'Scene modeling and statistical learning based robust pedestrian detection algorithm', *Acta Autom. Sin.*, 2010, **36**, (4), pp. 499–508

[3] Dalal, N., Triggs, B.: 'Histograms of oriented gradients for human detection'. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, San Diego, USA, June 2005, pp. 886–893

[4] Li, W., Dai, D., Tan, M., *et al.*: 'Fast algorithms for linear and Kernel SVM + '. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Las Vegas, USA, June 2016, pp. 2258–2266

[5] Maji, S., Berg, A.C., Malik, J.: 'Classification using intersection kernel support vector machines is efficient'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Anchorage, USA, June 2008, pp. 1–8

[6] Wu, J.: 'Efficient HIKSVM learning for image classification', *IEEE Trans. Image Process. Publ. IEEE Signal Process. Soc.*, 2012, **21**, (10), pp. 4442–4453

[7] Ojala, T., Harwood, I.: 'A comparative study of texture measures with classification based on feature distributions', *Pattern Recognit.*, 1996, **29**, (1), pp. 51–59

[8] Tan, X., Triggs, B.: 'Enhanced local texture feature sets for face recognition under difficult lighting conditions', *IEEE Trans. Image Process.*, 2010, **19**, (6), pp. 1635–1650

[9] Mu, Y., Yan, S., Liu, Y., *et al.*: 'Discriminative local binary patterns for human detection in personal album'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Anchorage, USA, June 2008, pp. 1–8

[10] Heikkilä, M., Pietikäinen, M., Schmid, C.: 'Description of interest regions with local binary patterns', *Pattern Recognit.*, 2009, **42**, (3), pp. 425–436

[11] Papageorgiou, C., Poggio, T.: 'A trainable system for object detection', *Int. J. Comput. Vis.*, 2000, **38**, (1), pp. 15–33

[12] Walk, S., Majer, N., Schindler, K., *et al.*: 'New features and insights for pedestrian detection'. Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, San Francisco, USA, June 2010, pp. 1030–1037

[13] Dollár, P., Tu, Z., Perona, P., *et al.*: 'Integral channel features'. Proc. British Machine Vision Conf., London, UK, September 2009, pp. 1–11

[14] Wang, X.: 'An HOG-LBP human detector with partial occlusion handling'. Proc. Int. Conf. Computer Vision, Kyoto, Japan, September 2009, pp. 32–39

[15] Dollár, P., Belongie, S., Perona, P.: 'The fastest pedestrian detector in the west'. Proc. BMVC, Aberystwyth, UK, September 2010, pp. 1–11

[16] Benenson, R., Mathias, M., Timofte, R., *et al.*: 'Pedestrian detection at 100 frames per second'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Providence, USA, June 2012, pp. 2903–2910

[17] Cheng, M.M., Zhang, Z., Lin, W.Y., *et al.*: 'Bing: binarized normed gradients for objectness estimation at 300fps'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Columbus, USA, June 2014, pp. 3286–3293

[18] Mao, J., Xiao, T., Jiang, Y., *et al.*: 'What can help pedestrian detection?'. Proc. IEEE Conf. Computer Vision Pattern Recognition, Honolulu, USA, July 2017, pp. 6034–6043

[19] Redmon, J., Divvala, S., Girshick, R., *et al.*: 'You only look once: unified, real-time object detection'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Las Vegas, USA, June 2016, pp. 779–788

[20] Chi, J.T., Chi, E.C., Baraniuk, R.G.: 'K-pod: a method for k-means clustering of missing data', *Am. Stat.*, 2016, **70**, (1), pp. 91–99