Northumbria Research Link

Citation: Wang, Lei, Zhang, Baochang, Han, Jungong, Shen, Linlin and Qian, Cheng-shan (2016) Robust object representation by boosting-like deep learning architecture. Signal Processing: Image Communication, 47. pp. 490-499. ISSN 0923-5965

Published by: Elsevier

URL: http://dx.doi.org/10.1016/j.image.2016.06.002 <http://dx.doi.org/10.1016/j.image.2016.06.002>

This version was downloaded from Northumbria Research Link: http://nrl.northumbria.ac.uk/28492/

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: http://nrl.northumbria.ac.uk/policies.html

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)

www.northumbria.ac.uk/nrl

northumbria
UNIVERSITY NEWCASTLE

Robust Object Representation by Boosting-like Deep Learning Architecture

Lei Wang[1], Baochang Zhang[1], Jungong Han[2], Linlin Shen[3], Cheng-shan Qian[4]
[1]School of Automation Science and Electrical Engineering
Beihang University, Beijing, China
[2] Computer Science and Digital Technologies Department, Northumbria University, Newcastle, UK
3Computer Vision Institute, School of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
4 School of Computer & Software, Nanjing University of Information Science & Technology, China

Abstract. This paper presents a new deep learning architecture for robust object representation, aiming at efficiently combining the proposed synchronized multi-stage feature (SMF) and a boosting-like algorithm. The SMF structure can capture a variety of characteristics from the inputting object based on the fusion of the handcraft features and deep learned features. With the proposed boosting-like algorithm, we can obtain more convergence stability on training multi-layer network by using the boosted samples. We show the generalization of our object representation architecture by applying it to undertake various tasks, i.e. pedestrian detection and action recognition. Our approach achieves 15.89% and 3.85% reduction in the average miss rate compared with ACF and JointDeep on the largest Caltech dataset, and acquires competitive results on the MSRAction3D dataset.

Key words: boosting, deep learning, object representation, synchronized feature

1 Introduction

Existing pattern recognition methods generally include two major steps: feature extraction and classifier design. The quality of visual features is crucial for a wide range of computer vision topics, e.g., scene classification, object detection and human action recognition. Handcrafted feature and learning based feature are two commonly used features feeding into a decision-making algorithm. Handcrafted features are usually inspired by the domain and the particular application, intending to capture certain morphological, statistical or texture attributes of objects. Therefore, the rules of handcrafted feature extraction are varied according to different application fields. Many famous handcrafted features, such as HOG [1], Haar-like [2], SIFT [3], covariance descriptors [4], integral channel features [5] and 3D geometric characteristic feature [6], have been successfully applied in pedestrian detection. Moreover, the spatio-temporal features [7, 8, 9, 10] and depth images-based features [11, 12, 13] are widely utilized for human action recognition. Differently, the learning-based features are captured with data-driven tending to be domain agnostic. These feature extraction methods attempt to automatically obtain intrinsic representations from training samples, which generally learn additional feature bases that cannot be represented through any of the handcrafted features. Sermanet *et al.* [14] initially propose ConvNet using the original pixel values as the input to train multi-stage automatic sparse convolution encoder by combining unsupervised and supervised methods. ConvNet shows an impressive result on the INRIA pedestrian database. UDN [15] is a deep neural network model jointly combined with the deformation and occlusion models. The approach achieves a new state-of-the-art result by giving a lower miss rate on Caltech and ETH dataset than other existed methods. In [16], deep features learned by CNN are extracted from training samples and integrated with support vector machine [17, 18] to obtain a higher performance on the pedestrian detection.

From the above brief reviews of previous work, it is clear that the handcrafted feature extraction methods are specifically designed for the particular application or the domain. Although it has superior description ability for the textural objects, it cannot learn the intrinsic characteristic of the signal and is generally with a poor adaptability. In contrast, the learned features can overcome these weaknesses. However, they normally require a large number of training samples and are expensive in terms of the computational load and the hardware cost. Therefore, a straightforward idea is to fuse these two features, which is naturally a promising way toward more robust and adaptable representations. However, to our best of knowledge, few works are done along this direction. In this paper, the low-level handcrafted feature is used as the input of the networks to learn higher representations. In addition, the convergence speed and stability are two other major problems in the deep networks with a propagation feedback scheme. Existing methods [19, 20] fail to solve these issues. In order to address the above problems, we propose a new deep learning framework, which only requires a small-scale CNN but achieves higher performance with less computational costs. The proposed synchronized multi-stage feature (SMF) structure can lead to an ensemble of the heterogeneous models, thus speeding up the training process and making our model more powerful by deeply investigating into the diversity among different features. Moreover, the proposed boosting-like algorithm gradually tunes the sample weights in the feedback propagation, thereby gaining more stability and performance improvement. Based on convolutional neural network (CNN) [21], our proposed deep structure contains two convolutional layers to obtain higher-level feature representations. The final classifier is only a simple single neural network. The performance of this framework is verified by two challenging applications: pedestrian detection and action recognition.

The contributions made in this paper are three-fold: firstly, a feature fusion scheme is proposed to extract more intrinsic and adaptive features, which provides a feasible way of combining the traditional handcrafted features and newly invented deep learned features. Secondly, a synchronized multi-stage features feed-forward structure is introduced into deep neural network, which can capture different kind of structure information, and thus improve the network convergence speed. Thirdly, the boosting-like algorithm helps to generate more stable and effective deep networks. Revealed by the classification results, the weights of the training samples in the feedback propagation are gradually adjusted to improve the system stability and the classification performance.

The rest of this paper is organized as follows. In section 2, our deep convolution framework is described. In section 3, the proposed boosting-like algorithm is elaborated. In section 4, synchronized multi-stage feature structure is presented. The handcrafted features are concisely introduced in section 5. Experimental results are provided and analyzed in section 6. Finally, the paper is concluded in section 7.

2 Deep convolutional structure

CNN consists of sequentially placed pairs of convolutional and sub-sampling layers. The fundamental deep structure used in this paper is shown in Fig.1.

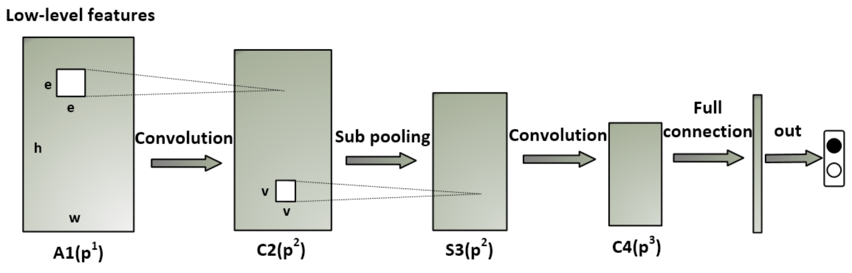


Fig.1. Overview of our fundamental deep convolutional structure

Let us now look at one example showing how the CNN works. At certain layer, the $w \times h \times p^1$ previous layer's features maps are convolved with $e \times e$ kernels $\omega_{i,j}$ and put through the activation fuction $f(\cdot)$ to form the $(w-e+1) \times (h-e+1) \times p^2$ output feature maps. For ease of explanation, the j-th feature map of the l-th layer M_j^l is given as

$$M_j^l = f\left(\sum_{i \in I_j} M_i^{l-1} * \omega_{i,j} + b_j^l\right), \tag{1}$$

where I_j denotes a selection of input maps. The sub-pooling can be represented as

$$S_j^l = f\left(\beta_j^l down(M_j^{l-1}) + b_j^l\right). \tag{2}$$

Here, each output map S_j^l is given its own multiplicative bias β_j^l and an additive bias b_j^l. The second convolutional layer $C4$ is same to $C2$ except for the kernel size. The classifier should be differentiable with respect to weights so that we can employ the back propagation algorithm to train the network.

In this paper, $C2$ uses sigmoid and $C4$ uses hyperbolic tangent. The kernels in $C2$ layer own the same size, but they are different in the $C4$ layer. We adopt mean-pooling with $\beta_j^l = 1/4$. The final classifier is a simple single neural network. When we use Caltech dataset to train our network, the visualization of the kernels and feature maps are shown in Fig.2 and Fig.3.

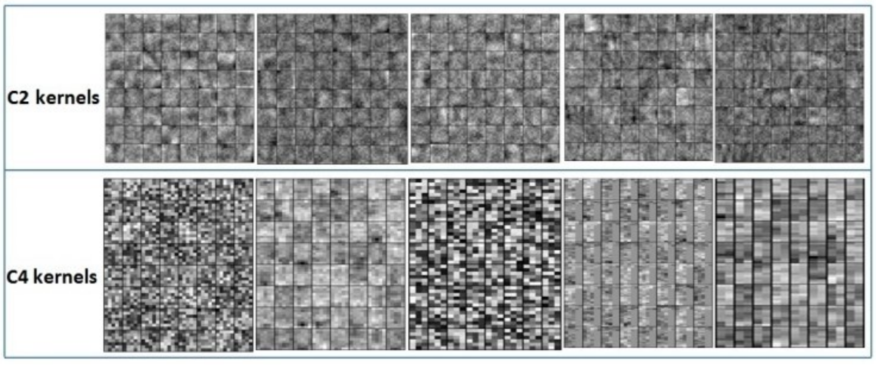


Fig.2. Visualization of convolution kernels for C2 and C4 layers on Caltech-train

In Fig.2, the visual maps of kernels in $C2$ mainly contain edges information as shown in the first row. However, the visual maps of kernels in $C4$ mainly include the corner information as shown in the second row. It demonstrates that deep network can acquire higher-level essential and detailed features with layers increasing to represent more intrinsic characteristics of objects.

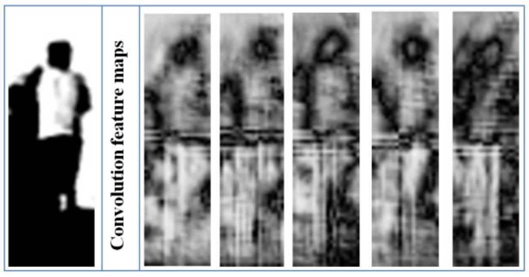


Fig.3. Visualization of convolution feature maps for *C2* and *C4* layers on Caltech-train

In Fig.3, one single grayscale image of pedestrian is used as the input channel, and we visualize the $C2$ convolution feature maps. From the results we can see that it contains more details than manual features in Fig.6.

3 Boosting-like deep learning algorithm

Multi-layer and feed-forward CNN trained with the back propagation (BP) algorithm are by far the most commonly used architecture in the literature. There is a trade-off between the convergence stability and speed of the training algorithm. The standard BP algorithm shows a very slow rate of convergence and a high dependency on the value of the learning rate parameter [22]. The learning rate should be sufficiently large to allow a fast convergence. However, the large value can result in falling into local minimum and causing oscillation [23]. To cope with the problem, we propose the boosting-like deep learning algorithm (BDL) shown in Fig.4.

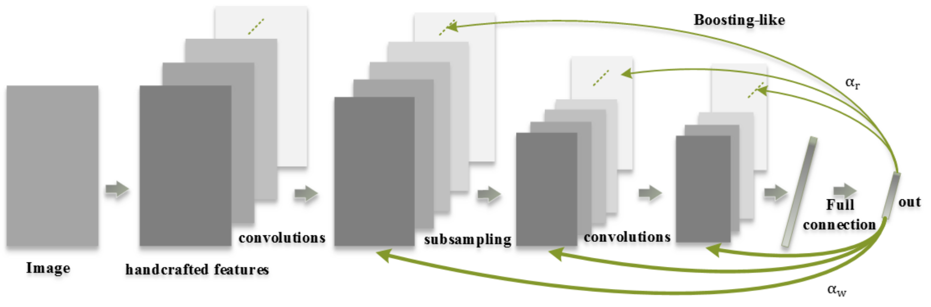


Fig.4. The convolutional structure with boosting-like algorithm, it shows how the algorithm operates in the deep network.

We first review the derivation by traditional neural network with the back propagation algorithm. Generally, mean squared error function is used as the objective function of CNN. For a multiclass problem with c classes, the n-th sample error is expressed as

$$E = \frac{1}{2}\sum_{k=1}^{c}(o_k^n - y_k^n)^2. \tag{3}$$

Here o_k^n is the k-th output layer unit, and y_k^n denotes the label of k-th dimension of the n-th sample. Assuming that the input sample is multiplied by a penalty weight α, the input u^l of the l-th layer of the $(l-1)$-th layer is subjected to

$$u^l = \alpha w^l x^{l-1} + b^l\,,\quad x^l = f(u^l)\,. \tag{4}$$

Here, w^l denotes the weight of the l layer and b^l refers to the bias. x^{l-1} is the output of the (*l-1*) layer, and $f(\cdot)$ is the excitation function. Then we have

$$\delta^l = f'(u^l)(y^n - o^n) . \tag{5}$$

The derivative of E against w^l is shown as follows

$$\frac{\partial E}{\partial w^l} = \delta^l \frac{\partial u}{\partial w^l} = x^{l-1} f'(u^l)(y^n - o^n)\alpha. \tag{6}$$

Finally, the delta updating rule is applied to each neuron to gain new weights given by

$$w^{l+1} = w^l - \eta x^{l-1} f'(u^l)(y^n - o^n)\alpha. \tag{7}$$

Here, η is the learning rate. Actually, the convolution neural network itself can be regarded as different level of cascaded feature extractors. The extracted features are changed from low level to high-level, and all results have a mutual complimentary. The output of one layer links to both previous layer and its successive layer. According to Eq. (7), we redistribute the feedback weights of the correctly and mistakenly classified samples, and feedback propagated them from the last layer to the starting layer.

$$(\delta^l)^{\tau+1} = ((o^n)^{\tau} - (y^n)^{\tau})\alpha^{\tau}, \tag{8}$$

where $(o^n)^{\tau} - (y^n)^{\tau}$ is the output error and it is also the output sensitivity in our CNN, $(o^n)^{\tau}$ is the actual output of the network, and $(y^n)^{\tau})$ is the target. α^{τ} is the penalty coefficient of correct and wrong classified samples. When the sample output value differs with its label, the penalty weight tends to be increased. Otherwise, it is decreased. Basically, this idea is similar to the boosting concept, which trains different classifiers through updating the weights of training samples to avoid over-fitting. The boosting-like way in deep learning can make the network performance more stable.

The solution of α is crucial to our method. According to new classification results of samples, there will be a new weights distribution D_τ of samples. In order to optimize the error function, boosting is integrated into the network calculation at the training stage, which not only strengthens the convergence stability, but also improves the performance. The details of our solution are formulated as follows. The distribution of samples weights is initialized as

$$D_1 = (d_{11}, \cdots d_{1i}, \cdots, d_{1N}),\ \ d_{1i} = \frac{1}{N}, i = 1,2,\cdots, N . \tag{9}$$

For the iteration $\tau = 1,2,\cdots,T$, we use samples with distribution D_τ to train the CNN classifier $G_\tau(x)$. The classification error rate e_τ on the training dataset is calculated as

$$e_\tau = P(G_\tau(x_i) \neq y_i) = \sum_{i=1}^{N} d_{\tau,i} I(G_\tau(x_i) \neq y_i). \tag{10}$$

Here, $I(x,y)$ is the indicative function, and $d_{\tau,i}$ is the weight of the i-th sample. The classification performance β_τ is calculated as

$$\beta_\tau = \frac{1}{2} ln \frac{1-e_\tau}{e_\tau} . \tag{11}$$

Here, β_τ is the inverse function of the variable e_τ indicating the classified performance. The new distribution $D_{\tau+1} = \left(d_{\tau+1,1}, \cdots d_{\tau+1,i}, \cdots, d_{\tau+1,N}\right)$ is shown as

$$N_\tau = \sum_{i=1}^{N} d_{\tau i} exp(-\beta_\tau y_i G_\tau(x_i)) \tag{12}$$

$$d_{\tau+1,i} = \frac{d_{\tau i}}{N_\tau} exp\left(-\beta_\tau y_i G_\tau(x_i)\right), i = 1,2,\cdots, N , \tag{13}$$

where N_τ denotes the1-norm normalized factor. During the training process, the new sample weight distribution $D_{\tau+1}$ is used to update the parameter α shown as follows

$$\alpha^{\tau+1} = 1 + d_{\tau+1,i}. \tag{14}$$

From Eq. (14), it can be deduced that α decreases when the sample is correctly classified as we expected. On the contrary, it increases when the sample is wrongly classified.

4 Synchronized multi-stage features

Multi-stage features are generated by a concatenating the outputs at different stages in the network to enrich features. In this paper, in order to extract more discriminative features of the sample with different views, we propose a synchronized multi-stage feature structure (SMF) through combining the multi-stage structure with the convolutional connection mode. Input images/features from different channels are separately and synchronously convolved with corresponding kernels shown in Fig.5. Through sub-sampling and cascading, the middle output is used as the first-stage feature. The output of the second convolutional layer is used as the second-stage feature, as shown in Fig.5. Since we feed various features into the networks separately, we need to align them after each layer in order to concatenate them properly. That interprets why we call our feature as a sort of synchronized feature. In general, SMF is able to separately and synchronously capture features of input channels with different meanings, and then to efficiently extract sufficient representations at different stages. In addition, we employ Depth Motion Map (DMM) [12] on MSRAction3D [24] dataset as input images for human action recognition.

4.1 Depth motion map

The concept of DMM was initially introduced in [25]. The same approach was adopted in [12] while the procedure for generating DMM was modified to reduce the computational complexity. In this paper, we adopt the method introduced in [12] due to its computational efficiency. Specifically, given a depth video sequence with N frames, each frame in the video is projected onto three orthogonal Cartesian planes to generate three 2D projected maps corresponding to the front, side and top views, denoted by map_f, map_t and map_s. DMM are then generated as follows

$$DMM_{\{f,t,s\}} = \sum_{j=1}^{N-1} |map_{\{f,t,s\}}^{j+1} - map_{\{f,t,s\}}^{j}|, \tag{15}$$

where j is the frame index, the motion characteristics can be effectively captured by DMM. An example of the three DMM is shown in Fig.5 as the input channels.

4.2 SMF

The SMF structure is shown in Fig.5. Low-level handcrafted features maps $\{HF\_f,\ HF\_t,\ HF\_s\}$ are firstly extracted from the front, side, and top views $\{DMM_f, DMM_t, DMM_s\}$ on action dataset. The handcrafted feature maps of the three views are then convolved independently and synchronously with the corresponding kernels, as

$$\begin{cases} Conv_f = HF\_f * \omega_{i,f}, & i = 1, \cdots, l_1 \\ Conv_t = HF\_t * \omega_{i,t}, & i = l_1 + 1, \cdots, l_2 \\ Conv_s = HF\_s * \omega_{i,s}, & i = l_2 + 1, \cdots, l_3 \end{cases} . \tag{16}$$

The sub-sampling function $S(\cdot)$ is then applied on the convolution operator to get the first-stage features, which are further used to train the second-stage feature $C4$. The ultimate human action features F_{all} fed into the classifier are generated by

$$F_{all} = \{C4, S(Conv_f), S(Conv_t), S(Conv_s)\}. \tag{17}$$

The input images from different views are traditionally convolved with all kernels regardless of the input channel signification, and then they are forced to be accumulated as the next stage input, which largely mitigates the representation ability. Aiming at solving this problem, SMF improves the recognition accuracy, since the combination of heterogeneous features can extract more discriminative information. The SMF is further proved to quickly converge on the training process because of the sufficient features. The red dashed box in Fig.5 indicates the common structure of general input channels, such as pedestrian dataset.

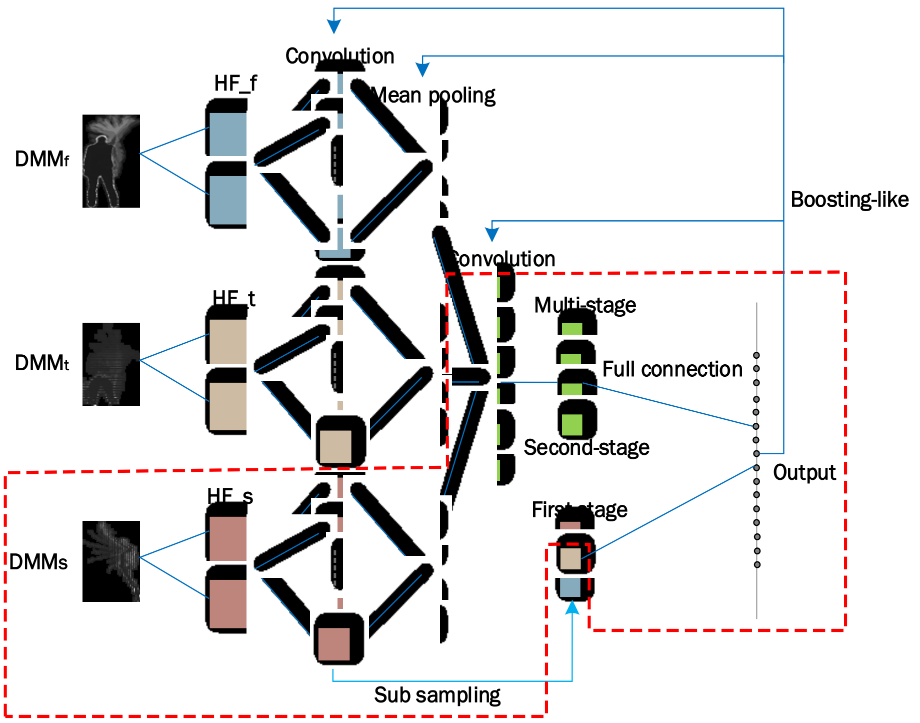


Fig.5. The synchronized multi-stage feature and boosting-like convolutional framework, it is the complete structure used in this paper.

It is worth mentioning that the boosting-like algorithm only changes the back propagation coefficient in the output layer of the deep network as introduced in section 3. Since the training method of our network is BP, this change can be automatically propagated back to all the previous layers in sequence shown in Fig.5.

5 Handcrafted features

The fusion of the traditional handcrafted features and deep learning features are introduced to extract robust features. According to different kinds of applications, various low-level handcrafted features are chosen as the input of network. They are then used as the input of the deep network to extract high-level features. In order to make this paper self-contained, two kinds of handcrafted features, i.e. aggregate channel features (ACF) and completed local binary patterns (CLBPs), are used for pedestrian detection and action recognition respectively. However, the framework can be applicable to any other different features.

5.1 Aggregate channel features for pedestrian detection

Channel features can be obtained through different linear/non-linear transformation of the input image I, such as Gabor filters and canny edge detection. Assuming the channel output is C , we get channel features $C = f(I)$, which denotes a simple first-order feature function such as the sum of pixels in a fixed rectangular area. High-order features can be calculated by combining several first-order functions via a variety of strategies.

In this paper, we extract low-level channel features in order to acquire superior. Firstly, we change the RGB input image into LUV color image; secondly, the gradient magnitude channel $|G|$ is calculated from the LUV image; and thirdly, six channels of histogram of gradient oriented $G_1 \sim G_6$ are gained through conversion processing on input images.

$$G_\theta(x,y) = G(x,y) \cdot I[\vartheta(x,y) = \theta] \tag{18}$$

Here, $G(x,y)$ is the gradient magnitude. Through multiplying with indication function, we get gradient oriented channels $G_\theta(x,y)$. For pedestrian images influenced seriously by illumination, the data in each channel is processed to be zero mean and unit variance. Since our network activation function is Sigmoid, this processing can also increase the convergence rate in the gradient descent process [26]. Fig.6 shows the visualizations of ACF. The first column is the original input image, the second to the tenth column correspond to LUV, $|\mathrm{G}|$ and the $\mathrm{G}_1 \sim \mathrm{G}_6$ channel features. The first row is the original channel features, and the second row is the one after normalization. The results show that regularization not only enhances the image resolution significantly, but also highlights the details of pedestrian.

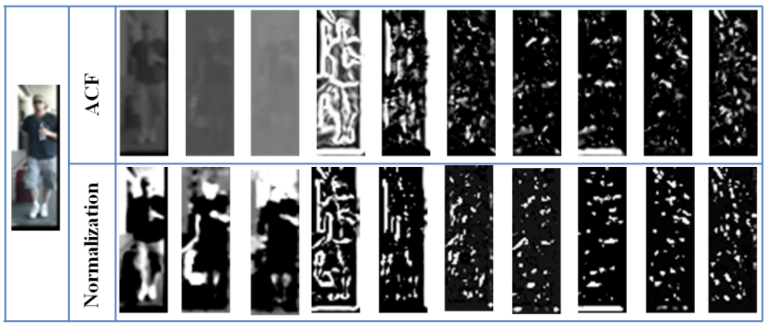
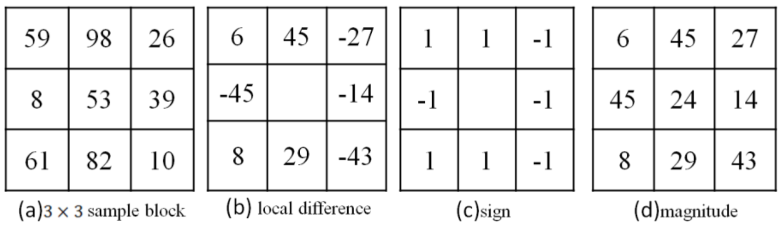


Fig.6. Comparison of channel features

5.2 Completed local binary patterns (CLBPs) for action recognition

LBP has been applied to various image classification applications, e.g. [20, 27, 28]. The traditional LBP operator only considers the sign information of the local difference vector. Therefore, structures with different values in magnitude may result in the same LBP codes. Aiming at addressing this problem, [20] proposes a more efficient local description operator completed LBP (CLBP), which simultaneously considers the local difference sign and magnitude. A local region is represented by its center pixel and a local difference sign-amplitude transforming (LDSMT). Fig.7 shows an example. The center pixels represent the image gray level and they are converted into a binary code by global thresholding namely CLBP-Center (CLBP_C). LDSMT decomposes the image local differences into two complementary components: the signs and the magnitudes, and two operators, namely CLBP-Sign (CLBP_S) and CLBP-Magnitude (CLBP_M), are proposed to code them. Especially, the traditional LBP is equivalent to the CLBP_S part of CLBP.



59	98	26
8	53	39
61	82	10

(a)3 × 3 sample block

6	45	-27
-45		-14
8	29	-43

(b) local difference

1	1	-1
-1		-1
1	1	-1

(c)sign

6	45	27
45	24	14
8	29	43

(d)magnitude

Fig.7. Samples of LDMST

Given a pixel in the image, $\mathrm{CLBP\_S}_{\mathrm{P,R}}$ operator is computed by comparing it with its neighbors shown as follows

$$CLBP\_S_{P,R} = \sum_{p=0}^{P-1} 2^p s\left(g_p, g_c\right), \tag{19}$$

$$s(x,y) = \begin{cases} 1, & x \geq y \\ 0, & x < y \end{cases}. \tag{20}$$

Here, g_c is the gray value of the central pixel, and $\mathrm{g_p}$ is the value of its neighbors. P is the total number of involved neighbors, and R is the radius of the neighborhood. Since the CLBP_M components are of continuous values instead of the binary values, they cannot be directly coded as that of CLBP_S. In order to code CLBP_M in a consistent format with CLBP_S, it is defined as follows

$$CLBP\_M_{P,R} = \sum_{p=0}^{P-1} 2^p s\left(m_p, c\right), \tag{21}$$

where m_p is the local absolute difference, c is an adaptive threshold. Here we set c to be the mean value of the whole local absolute difference image. $CLBP\_C$ operator is calculated as follows

$$CLBP\_C_{P,R} = s(g_c, av). \tag{22}$$

g_c is the gray value, and the threshold av is set as the mean value of the whole image. In order to reduce the computation, we only extract $\mathrm{CLBP\_S}_{\mathrm{P,R}}$ and $\mathrm{CLBP\_M}_{\mathrm{P,R}}$ low-level feature channels in this paper. The same normalized method as section 5.1 is adopted. A visualization example of CLBP feature and its normalization are shown in Fig. 8.

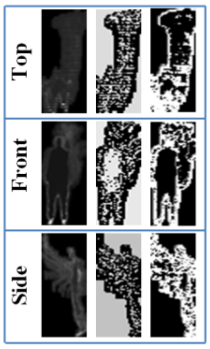


Fig.8. CLBP_M (middle) and CLBP_S (right) coded images corresponding to top, front and side in DMM (left) of a *high wave* depth sequence.

6 Experiment results and discussion

In this section, the proposed method (SMF-BDL) is evaluated on two different challenging tasks: Caltech pedestrian dataset [29] and MSRAction3D dataset [24]. The Caltech-test is the largest one among commonly used pedestrian datasets. It covers diverse complicated scenes including occlusion, illumination, deformation, and so on. While set00~set05 are used to train our model containing about 60000 training samples, set06~set10 are adopted as test sets. Usually sliding windows methods are used in detection stage. But it is well known that the feedback propagation in the deep learning network takes a lot of time. Therefore, we adopt HOG+CSS and linear SVM [15] to prune candidate detection windows for efficiency. These candidate windows have a high recall rate, but at the same time contain a lot of false positive windows. Moreover, MSRAction3D dataset is captured by commercial depth camera which includes 557 action sequences with 240×320 resolution. The depth motion maps (DMM) [12] on MSRAction3D are utilized as the input images.

6.1 Caltech dataset

The evaluation criterion proposed in [29] is adopted to check detection performance of our framework, which demonstrates the log-average miss rate versus false positive rate per image (FPPI). In the experiment, we evaluate the detection performance using the subset with pedestrians more than 49 pixels in height and less than 35% in occlusion. We compare our method with the popular related approaches such as VJ [30], ConvNet [31], HOG [1], ACF [32], HOGCSS [15] and JointDeep [15]. These methods use various kinds of features, classifiers and deformation models. Our method is denoted by SMF-BDL.

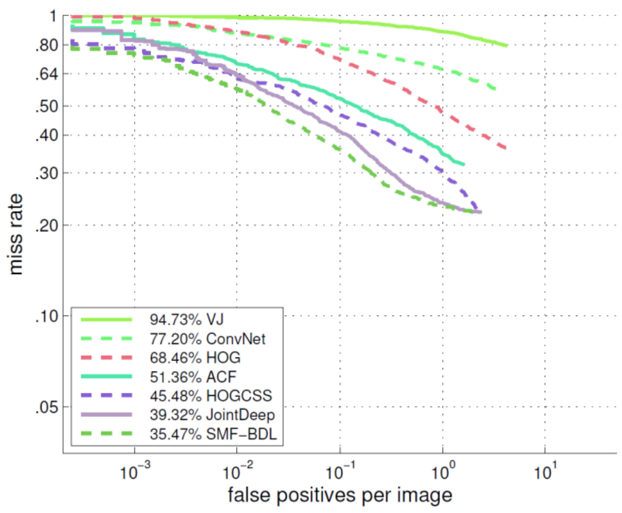


Fig.9. Comparison of log-average miss rate versus false positives per image (FPPI) between our method (SMF-BDL) and the related methods on Caltech dataset.

Fig.9 shows that the average miss rate of our method (SMF-BDL) is 35.47%. ACF is the low-level handcrafted channel features used in our network input, and HOGCSS is the method we used to prune candidate windows. Our method (SMF-BDL) gets 15.89% and 10.01% performance gains compared with the ACF and HOGCSS, respectively. Our method also performs 3.85% better than JointDeep, which also uses low-lever channel features and CNN. It should be noted that there are deformation and visibility reasoning layer in Jointdeep. Fig.10 shows some samples of pedestrian detection results on Caltech dataset using our SMF-BDL method and ACF.

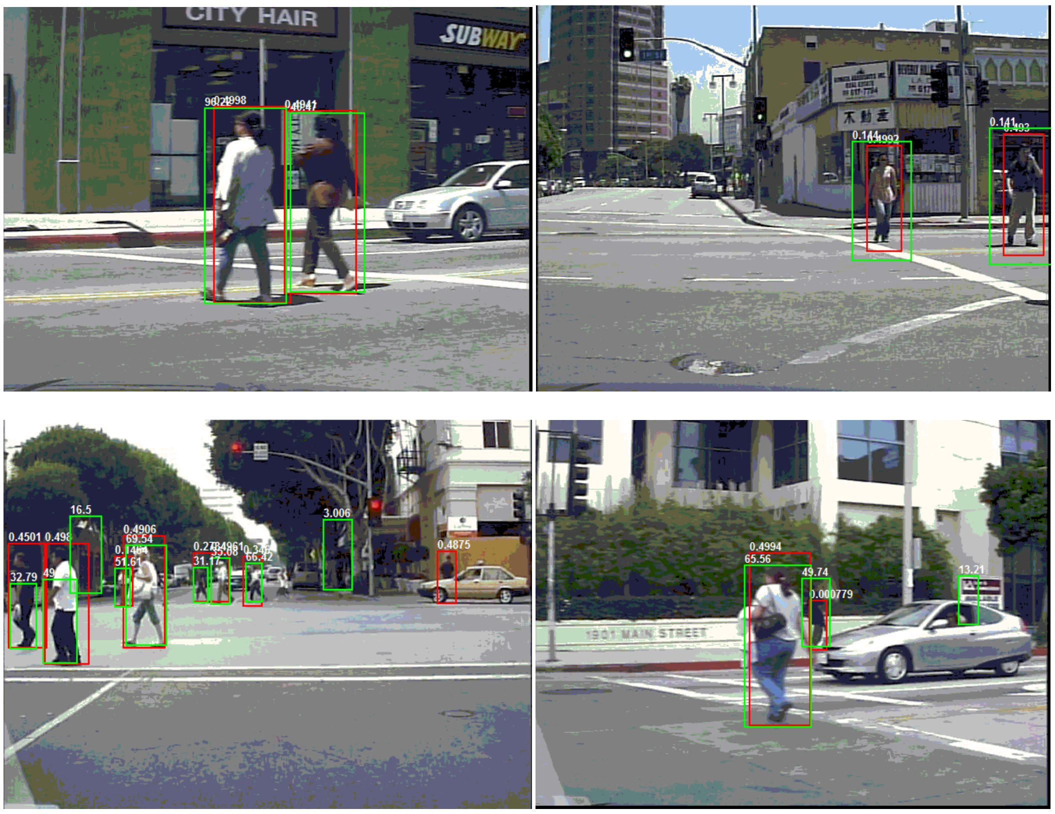


Fig.10. Detection examples on Caltech dataset, where bounding boxes with red color represent our method (SMF-BDL), and green ones represent ACF.

6.2 MSRAction3D

The dataset contains 20 actions performed by 10 persons, and each action is repeated 2~3 times by each person, including horizontal wave, beat, racket and so on. In order to compare with other state-of-the-art methods, we adopt experimental setting reported in [24]. The actions are divided into three subsets AS1, AS2 and AS3, and each subset contains eight actions. There are three test strategies:

Test 1: For each subject, 1/3 samples are used for training and the rest for testing.

Test 2: For each subject, 2/3 samples are used for training and the rest for testing.

Test 3: The subjects with odd number (1, 3, 5, 7, 9) are used for training, and the rest for testing.

In order to show the impact of different structures in **SMF-BDL** on system performance, we perform all of the three tests. Table 1 reports the experimental results of four methods. **MS** refers to the method that only adopts the multi-stage structure without neither synchronous operation nor boosting-like algorithm. **SMF** indicates that we use the synchronized multi-stage features structure without applying boosting-like algorithm. **BDL** denotes the approach that only employs the boosting-like algorithm without SMF. It is worth noting that these four methods are all designed based on the same fundamental deep structure elaborated in section 2. Table 1 demonstrates that **MS** achieves the least accuracy gains, while **SMF** largely improves the system accuracy average by 3%. The boosting-like algorithm also has a positive effect on accuracy and primarily enhances the system stability. The **SMF-BDL** we proposed achieves the superior performance on all the three settings.

Table 1. Comparison of recognition accuracies (%) of different structures on MSRAction3D dataset

method	Test 1(average)	Test 2(average)	Test 3(average)
MS	93.6	94.3	86.3
SMF	96.7	98.7	89.1
BDL	94.1	95.6	87.6
SMF-BDL	97.3	99.1	90.8

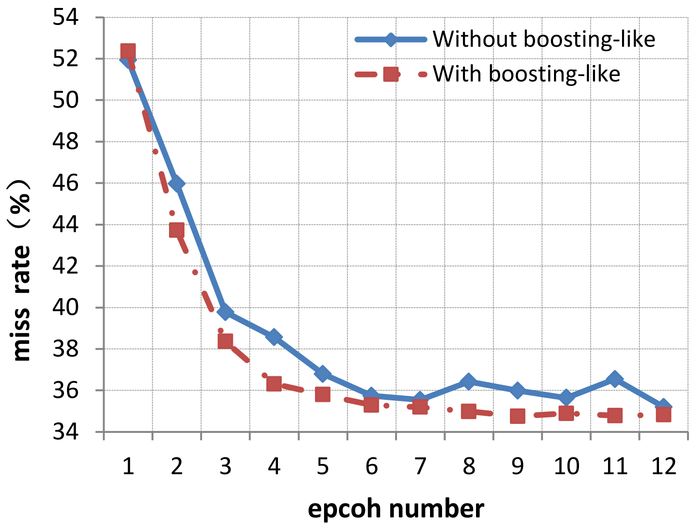
Table 2 shows the accuracy comparison of our method with the existing methods. It shows that the result of our method is highly competitive, achieving superior accuracies on both Test 1 and Test 2. Although DMM-HOG [25] is also based on DMM, its accuracies are 1.5% and 1.7% lower than that of our method on Test1 and Test2, respectively. But the advantage is not significant on Test 3, probably because the small training sets are hard to cover all the variations across different subjects. As we know, deep structure needs abundant samples to be sufficiently trained.

Table 2. Comparison of accuracies (%) our method SMF-BDL with other public methods on MSRAction3D dataset

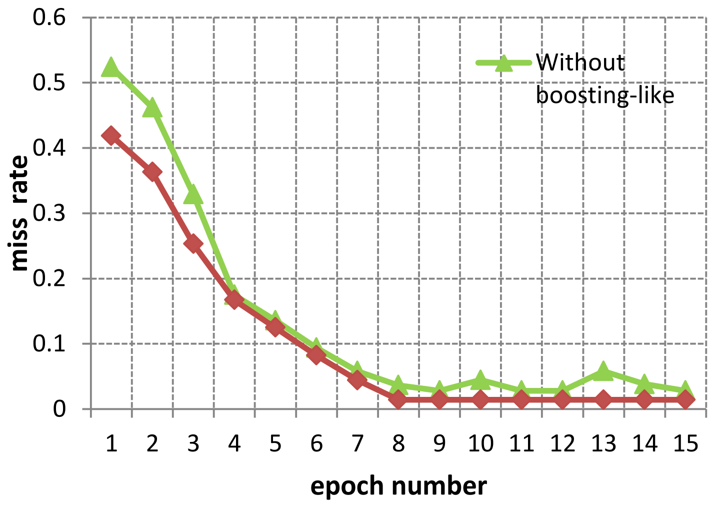
Method	Test 1				Test 2				Test 3			
	AS1	AS2	AS3	Average	AS1	AS2	AS3	Average	AS1	AS2	AS3	Average
Li et al.[24]	89.5	89.0	96.3	91.6	93.4	92.9	96.3	94.2	72.9	71.9	79.2	74.7
DMM-HOG[25]	97.3	92.2	98.0	95.8	98.7	94.7	98.7	97.4	96.2	84.1	94.6	**91.6**
HOJ3D[33]	98.5	96.7	93.5	96.2	98.6	97.2	94.9	97.2	88.0	85.5	63.6	79.0
Space-Time[34]	98.2	94.8	97.4	96.8	99.1	97.0	98.7	98.3	91.7	72.2	98.6	87.5
SMF-BDL	97.3	98.0	96.6	**97.3**	98.6	98.7	100	**99.1**	96.2	84.1	92.0	90.8

6.3 Stability analysis

In order to prove the effect on the stability using the proposed boosting-like algorithm, we evaluate it on both datasets. Fig.11 (a) shows the performances of the proposed algorithm with/without using boosting-like algorithm for pedestrian detection. The x-coordinate denotes training epochs, and the y-ordinate indicates average miss rate on Caltech-test, which is evaluated by protocol in [29] as well. Fig.11 (b) shows the similar results on MSRAction3D.



(a) Caltech dataset



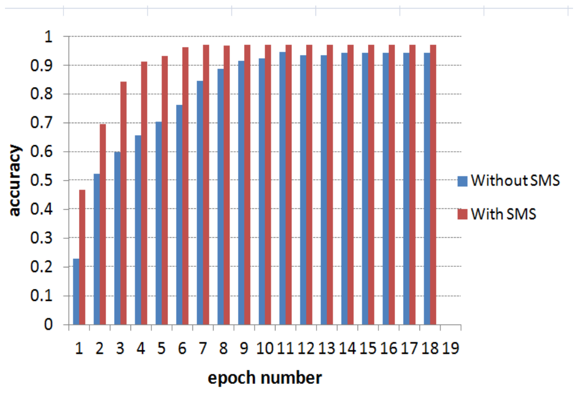
(b) MSRAction3D dataset

Fig.11. Comparison of Boosting-like in terms of stability and detection performance

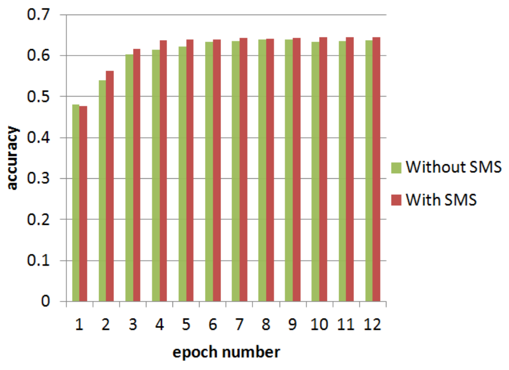
Fig.11 (a) shows that: firstly, the oscillation is smaller and the curve is relatively stable when using boosting-like algorithm in the feedback propagation. On the contrary, system stability is poor without using the boosting-like method. Secondly, the boosting-like algorithm achieves 0.48% performance gain on pedestrian detection. The same conclusion holds for the MSRAction3D as shown in Fig.11 (b), and 0.15% miss rate improvement is achieved by integrating the boosting-like algorithm. Therefore, the boosting-like algorithm not only improves the network stability without reducing convergence speed, but also slightly improves the detection accuracy.

6.4 Convergence speed

In this section, experiments are carried out to test the convergence speed of boosting-like algorithm. Fig.12 shows the convergence speed of the algorithm with/without SMF using the MSRAction3D data set. From Fig.12 (a) we can see that the accuracy of SMF is basically stable after the 6th epoch, whereas it does not stabilize until the 13th epoch when SMF is not used. At the same time, the recognition accuracy of the proposed algorithm with SMF is 2.8% higher than that of the algorithm without SMF, which significantly increases the convergence speed, probably because features are sufficient to train a superior classifier. In addition, the same conclusion holds for the pedestrian detection from Fig.12 (b).



(a) MSRAction3D dataset



(b) Caltech dataset

Fig.12. Convergence speed comparison of whether or not using SMF on MSRAction3D dataset and Caltech dataset. The horizontal axis denotes the epoch number, and the vertical axis is the accuracy.

Apart from testing the convergence speed of the algorithm, we also measure the running-speed of the entire system on the application of pedestrian detection. The training for 70k samples with the resolution of 84x28 at each epoch spends roughly 35 minutes, which implies that we only need 0.03 second to train each sample. On the test side, our algorithm has detected pedestrians on 11k images within 5 minutes, which is far more than a real-time system. All the measurements are carried out based on a Laptop PC (I5 CPU, 8G RAM) using matlab programming.

7 Conclusion

This paper proposes an effective deep detection framework named SMF-BDL. This model merges handcrafted method via a learning method to extract more superior features. The SMF structure is introduced, which can effectively deal with the recognition problems of input channels with different meanings and improve convergence speed. Due to the limited training samples, over-fitting is a major problem of a deep neural network. At the same time, the convergence stability of the network is also a challenge. Based on this SMF structure, we propose the boosting-like algorithm to prevent over-fitting and improve the system stability. It adjusts updating-rate according to the classification conditions of samples in the training process. At last, our method achieves superior performance on pedestrian detection and competitive results on action recognition. In the future, we plan to use MKL algorithm to optimize the weights attached to different views, which will definitely improve the system performance [35][36]. Furthermore, we will also apply the SMF-BDL for other recognition or detection tasks.

References

[1] T. Watanabe, Ito S, Yokoi K. Co-occurrence. Histograms of Oriented Gradients for Human Detection[J]. IMT, 2010, 5:39-47.

[2] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. IJCV, 2005, 63(2):153-161.

[3] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. IEEE 12th International Conference, 2009:606-613.

[4] O Tuzel, F Porikli, P Meer. Pedestrian detection via classification on riemannian manifolds. IEEE Trans. PAMI, 2008, 30(10):1713-1727.

[5] P. Doll´ar, Z. Tu, P. Perona, S. Belongie. Integral channel features. In BMVC, 2009, 2.

[6] Hoiem D., Efros A. A., Hebert M. Putting Objects in Perspective[J]. International Journal of Computer Vision, 2008, 80(1):2137-2144.

[7] Liu L, Shao L, Li X, et al. Learning Spatio-Temporal Representations for Action Recognition: A Genetic Programming Approach.[J]. IEEE Trans Cybern, 2015.

[8] Shao L, Zhen X, Tao D, et al. Spatio-Temporal Laplacian Pyramid Coding for Action Recognition[J]. Cybernetics IEEE Transactions on, 2014, 44(6):817-827.

[9] Zhu F, Shao L. Weakly-Supervised Cross-Domain Dictionary Learning for Visual Recognition[J]. International Journal of Computer Vision, 2014, 109(1-2):42-59.

[10] Jones S, Shao L. Unsupervised Spectral Dual Assignment Clustering of Human Actions in Context[C]// 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society, 2014:604-611.

[11] Chen C, Jafari R, Kehtarnavaz N. Improving Human Action Recognition Using Fusion of Depth Camera and Inertial Sensors[J]. Human-Machine Systems, IEEE Transactions on, 2015, 45(1):51-61.

[12] Chen C, Liu K, Kehtarnavaz N. Real-time human action recognition based on depth motion maps [J]. Journal of Real-Time Image Processing, 2013:1-9.

[13] Chen C, Jafari R, Kehtarnavaz N. Action Recognition from Depth Sequences Using Depth Motion Maps-based Local Binary Patterns [J]. IEEE Xplore, 2015:1092-1099.

[14] Sermanet P, Soumithchintala K, Lecun Y. Pedestrian Detection with Unsupervised Multi-Stage Feature Learning[J]. IEEE Conference on Computer Vision & Pattern Recognition, 2012: 3626-3633.

[15] Ouyang W, Wang X. Joint Deep Learning for Pedestrian Detection[C]// Computer Vision (ICCV), 2013 IEEE International Conference on. IEEE, 2013:2056-2063.

[16] Ke W, Zhang Y, Wei P, et al. Pedestrian detection via PCA filters based convolutional channel features[C]// Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE, 2015.

[17] Gu B, Sheng V S, Tay K Y, et al. Incremental Support Vector Learning for Ordinal Regression[J]. Neural Networks & Learning Systems IEEE Transactions on, 2015, 26:1403-1416.

[18] Bin Gu, Victor S. Sheng, Zhijie Wang, Derek Ho, Said Osman, Shuo Li, "Incremental learning for ν-Support Vector Regression," Neural Networks, (DOI: doi:10.1016/j.neunet.2015.03.013), 2015.

[19] P. Doll´ar, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2014.

[20] Ojala T. Multiresolution gray-scale and rotation invariant texture classification with local binary

patterns[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2002, 24(7):971-987.

[21] LeCun Y,Bottou L, Bengio Y,et al. Gradient-based learning applied to document recognition[J]. Proc of the IEEE, 1998, 86(11): 2278-2324.

[22] M. Moreira, E. Fiesler. Neural Networks with Adaptive Learning Rate and Momentum Terms[J]. Idiap, 1995.

[23] CHEN Y N, HAN C C, WANG C T, et al. The application of a convolution neural network on face and license plate detection[C]. Proc. 18th Int. Conf. Pattern Recognition (ICPR'06), 2006, 552-555.

[24] Li W., Zhang Z., Liu Z.. Action recognition based on a bag of 3D points[C]// 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops. 2010:9-14.

[25]Yang X., Zhang C., Tian Y. Recognizing actions using depth motion maps-based histograms of oriented gradients[J]. Acm International Conference on Multimedia, 2012:1057-1060.

[26] Bouvrie J. Notes on Convolutional Neural Networks[J]. Neural Nets, 2006.

[27] Li W, Chen C, Su H, et al. Local Binary Patterns and Extreme Learning Machine for Hyperspectral Imagery Classification[J]. Geoscience & Remote Sensing IEEE Transactions on, 2015, 53(7):3681-3693.

[28] Chen C, Zhang B, Su H, et al. Land-use scene classification using multi-scale completed local binary patterns[J]. Signal Image & Video Processing, 2015.

[29] Dollar P, Wojek C, Schiele B, et al. Pedestrian Detection: An Evaluation of The State Of The Art [J]. Pattern Analysis & Machine Intelligence IEEE Transactions on, 2012, 34(4):743-761.

[30] P. Viola, M. J. Jones, D. Snow. Detecting pedestrians using patterns of motion and appearance. IJCV, 2005, 63(2):153-161.

[31] Sermanet P, Kavukcuoglu K, Chintala S, et al. Pedestrian Detection with Unsupervised Multi-Stage Feature Learning[J]. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2012: 3626-3633.

[32] Appel R, Perona P, Belongie S. Fast Feature Pyramids for Object Detection [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2014, 36(8):1.

[33] Xia L., Chen C. C., Aggarwal J K. View invariant human action recognition using histograms of 3D joints[C], Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. IEEE, 2012:20-27.

[34] A. W. Vieira, E. R. Nascimento, G. L. Oliveira, Z. Liu, M. F. Campos. On the improvement of human action recognition from depth map sequences using space-time occupancy patterns. Pattern Recognition Letters, 2014, 36: 221-227.

[35] Liu X., Wang L., Yin J., Zhu E., Zhang J. An efficient approach to integrating radius information into multiple kernel learning. Cybernetics IEEE Transactions on, 2013, 43(2): 557-569.

[36] Liu X., Wang L., Zhang J., Yin J. Sample-adaptive multiple kernel learning. Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014: 1975-1981.