Applied Soft Computing 36 (2015) 218–227

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Human action recognition in H.264/AVC compressed domain using meta-cognitive radial basis function network

R. Venkatesh Babu [a,*], Badrinarayanan Rangarajan [b], Suresh Sundaram [b], Manu Tom [a]

[a] Supercomputer Education and Research Centre, Indian Institute of Science, Bangalore, India
[b] School of Computer Engineering, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Received 31 December 2013
Received in revised form 28 April 2015
Accepted 24 June 2015
Available online 23 July 2015

Keywords:
Human action recognition
Compressed domain
Motion vectors
Quantization parameters
PBL-McRBFN
Meta-cognitive learning

ABSTRACT

In this paper, we propose a H.264/AVC compressed domain human action recognition system with projection based metacognitive learning classifier (PBL-McRBFN). The features are extracted from the quantization parameters and the motion vectors of the compressed video stream for a time window and used as input to the classifier. Since compressed domain analysis is done with noisy, sparse compression parameters, it is a huge challenge to achieve performance comparable to pixel domain analysis. On the positive side, compressed domain allows rapid analysis of videos compared to pixel level analysis. The classification results are analyzed for different values of Group of Pictures (GOP) parameter, time window including full videos. The functional relationship between the features and action labels are established using PBL-McRBFN with a cognitive and meta-cognitive component. The cognitive component is a radial basis function, while the meta-cognitive component employs self-regulation to achieve better performance in subject independent action recognition task. The proposed approach is faster and shows comparable performance with respect to the state-of-the-art pixel domain counterparts. It employs partial decoding, which rules out the complexity of full decoding, and minimizes computational load and memory usage. This results in reduced hardware utilization and increased speed of classification. The results are compared with two benchmark datasets and show more than 90% accuracy using the PBL-McRBFN. The performance for various GOP parameters and group of frames are obtained with twenty random trials and compared with other well-known classifiers in machine learning literature.

1. Introduction

Human action recognition has been an area of keen interest to computer vision researchers for the past few decades. Day by day, length and breadth of the problem statement are expanding and researchers have come up with solutions to tackle scale, appearance, illumination, orientation variations, and occlusions. Speed of recognition has not been given ample thrust in the evaluations. The aim of this research is to highlight the speed of recognition as an important parameter in the analysis of action recognition algorithms.

Conventionally the problem of action recognition is tackled in pixel domain. This involves two major steps: decompressing the video and extracting low level features like optical flow. Further, decompression increases the data by 50–100 times, leading to a proportional increase in computational overhead. Whereas, compressed domain analysis utilizes the various compression parameters that are extracted by partial decoding for processing. These compression parameters such as motion vectors, quantization parameters etc. are low bit-rate data and require a fraction of computations/memory compared to the fully decoded pixel domain processing. On the other hand, since the compression parameters convey the approximate noisy information of the underlying pixel process, it is a huge challenge to achieve the performance comparable to pixel domain analysis. A substantial amount of research is being done in compressed domain owing to its high processing speed. A survey of proposed approaches for various applications in compressed domain is presented in [1].

Monitoring at high resolution, low bandwidth usage, reduced storage requirements, faster frame rates, and better video quality makes H.264/AVC [2,49] the ideal video standard for real-time applications. In a typical video transmission system, raw video input from the camera is compressed and streamed across a network targeting a decoder to reconstruct the source video. Bandwidth is the limiting factor over which bit-rate adjustment has an upper hand. Motion Vectors (MV) are the indication of how motion

* Corresponding author. Tel.: +91 80 22932900.
E-mail address: venky@serc.iisc.ernet.in (R. Venkatesh Babu).

http://dx.doi.org/10.1016/j.asoc.2015.06.054
1568-4946/

was compensated while exploiting the temporal frame redundancy during encoding. Quantization Parameter (QP) has a major role in regulating the bit rate during the encoding process. QP values can be adjusted to maintain a constant bit rate within the allowed channel bandwidth. Hence, real-time encoders heavily depend on varying the QP values to control the trade-off between video quality and compression. A combination of these features can be used for various applications of video analysis. For example, Biswas et al. [3] have proposed a crowd flow segmentation technique for H.264 compressed domain videos using motion vectors.

In the proposed approach, three types of features are extracted from the readily available quantization parameters and motion vectors of compressed H.264 bitstream. The first two features, Quantization-Parameter Gradient Image (QGI) spread and QGI projection profile, are obtained from the QGI at macro-block resolution (16×16). This QGI provides a compact representation of the action indicating the location and spatial-spread of the motion. The third feature, histogram of magnitude-weighted orientation of MVs, obtained at sub-macro-block resolution (4×4) provides important cues regarding the dynamics of the action. The main objective is to find the functional relationship between the features and action labels. One can employ machine learning approaches to estimate the decision boundaries.

Metacognitive Radial Basis Function Network (McRBFN) emulates the Nelson and Narens model of human meta-cognition [4], and has 2 components, namely, a cognitive component and a meta-cognitive component. Projection Based Learning algorithm of the Metacognitive Radial Basis Function Network (PBL-McRBFN) has a radial basis function network with Gaussian activation function at the hidden layer which is the cognitive component of McRBFN and a self-regulatory learning mechanism which is its meta-cognitive component. McRBFN begins with zero hidden neurons, adds and prunes neurons until an optimum network structure are obtained. The self-regulatory learning mechanism of McRBFN uses the best human learning strategy, namely, self-regulated learning to decide what-to-learn, when-to-learn and how-to-learn in a meta-cognitive framework. The meta-cognitive learning algorithm has been implemented in neuro-fuzzy [5–7], complex-valued [8,9] and neural networks [10]. Among these implementations, PBL-McRBFN [11,12] is computationally efficient and provides better generalization performance. Hence, PBL-McRBFN is used to approximate the functional relationship between the proposed features and action labels. The results show a superior performance of PBL-McRBFN in recognizing human actions. Recently it has been shown in neural network literature that learning algorithms employing meta-cognition (what-to-learn, when-to-learn and how-to-learn) provides better generalization performance [6,13–15].

The paper is structured in the following way. Related works on activity recognition in pixel and compressed domains are presented in Section 2. The proposed approach is discussed in detail in Section 3. Section 4 briefly explains the PBL-McRBFN classifier. Section 5 presents the experimental results and discussions. Concluding remarks and future works are stated in Section 6.

2. Related work

A lot of research has been reported till date in recognizing human actions in pixel domain. However, none of the algorithms in pixel domain have claimed recognition speed above 25 frames per second (fps). Applications demanding low memory utilization and faster speeds cannot employ those techniques because of the higher computational load. Pixel domain approaches [16–20] offer better classification performance at the cost of higher computational complexity and hence very few have achieved real-time recognition. Sadek et al. [21] presented a method for activity recognition, which works at 18 fps, using a finite set of features directly derived from different frames of action snippet. Yu et al. [22] reported a speed of 10–20 fps for action recognition by semantic texton forests which work on video pixels generating visual codewords. Li et al. [23] also claims real time accuracy by performing a luminance field manifold trajectory analysis based solution for human activity recognition.

Compressed domain approaches on the other hand, are more feasible for real-time applications owing to their faster performance. But, very less amount of research has been done in recognizing human actions in compressed video's. Ozer et al. [24] proposed a hierarchical approach for human activity detection and recognition in MPEG compressed sequences. Body parts were segmented out and Principal Component Analysis was performed on the segmented motion vectors prior to classification. Another notable work was put forward by Babu et al. [25] in MPEG compressed domain. The features extracted from motion vectors were fed to Hidden Markov Model (HMM) for classification. Babu et al. [26] later proposed a method using motion flow history and motion history image in MPEG compressed domain. Yeo et al. [27] proposed a high-speed algorithm for action recognition and localization in MPEG streams based on computing motion correlation measure by utilizing differences in motion direction and magnitudes. However, the algorithm cannot handle scale variations. The proposed method, on the other hand is robust to scale changes and attains slightly higher accuracy than [27]. Their method calculated optical flow from the MVs. After the formation of optical flow, the approach is equivalent to any pixel-based algorithm and hence is not a pure compressed domain approach.

In this paper, a novel algorithm is proposed for activity recognition using information from QPs, MB partition types, and MVs in the H.264/AVC compressed video. The gradient information of QP over the space is utilized to form QP Gradient Image (QGI) which provides vital clue regarding motion occurrence and the spread of the action. The features extracted from QGI and MVs are used to train the PBL-McRBFN classifier to recognize the actions. To our knowledge, this is the first work on action recognition in H.264/AVC [2] compressed domain.

3. QGI based PBL-McRBFN action recognition system

Overview of the proposed QGI based PBL-McRBFN classifier for action recognition approach is shown in Fig. 1. First, the QP delta (difference in QP values of adjacent macroblocks) and MV values are extracted by partial decoding of the input H.264 bitstream. QGI is then formed using the QP delta values which is then utilized to generate the QGI projection profile and spread features. Also, motion accumulated projection profile and histogram of magnitude-weighted orientation of motion vectors are formed as features using the extracted MVs [49]. All the individual features, after weighting, are combined to form the final feature vector. These feature vectors are used in PBL-McRBFN [11] to classify the video stream into action label.

3.1. Action representation and feature extraction

In this work, the action is represented in the following three ways: (i) Gradient Image, (ii) Motion Accumulation and (iii) Magnitude Weighted Orientation of Motion vectors (MWOM). Features extracted from these representations are used for classifying the actions.

3.1.1. Quantization parameter gradient image

H.264 allows to vary the value of QP on a macroblock (MB) basis to achieve better compression. Encoder maintains fixed QP values for MBs with low information and vice-versa. Adjacent MBs in raster-scan order will have very small change in QP values, if they

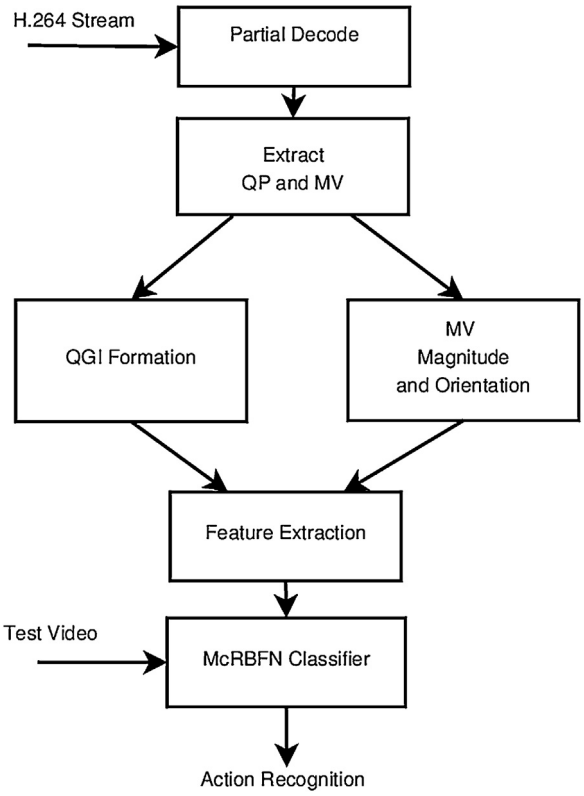

Fig. 1. Overview of the QGI based PBL-McRBFN action recognition system.

are part of the static background. Higher the delta value, higher is the edge information, which is a direct measure of action occurrence in the respective MB pair. The MBs present at the action area will have non zero QP-delta and will be classified as action-MBs. Spurious MBs with no action information can be removed by connected component analysis. This step is continued for all P frames and provides initial action segmentation with MB level precision.

To incorporate temporal content, a weighted accumulation of the filtered action-MBs over time is used. The requirement for the distribution of weights is that it should be maximum for the current frame and should symmetrically and monotonically decrease on both sides (forward and backward) as it goes farther away from this frame. Here we have chosen the weights as linearly decreasing with the current frame be given the weight 1.0. Forward (F_a) and backward accumulation factors (B_a) are used as the number of frames accumulated in forward and backward temporal directions respectively.

The effect of these parameters is given in the experiments' section. Even in real-time applications, buffering of frames is possible which validates the credibility of forward accumulation. The action-edge accumulated output for current frame $P_{acc}(T)$ is given by:

$$P_{acc}(T) = \sum_{k=0}^{B_a} w(T-k)P(T-k) + \sum_{k=1}^{F_a} w(T+k)P(T+k) \quad (1)$$

where $P(T-k)$ and $P(T+k)$ denotes the output of initial filtering stage for a frame which is k-frames farther from the current frame $P(T)$ in reverse and forward temporal directions respectively. Similarly, $w(T-k)$ and $w(T+k)$ denotes the respective weights assigned which are linearly distributed across past and future frames. To generate the final QGI for a bunch of frames, the action-edge MBs of each frame are again accumulated

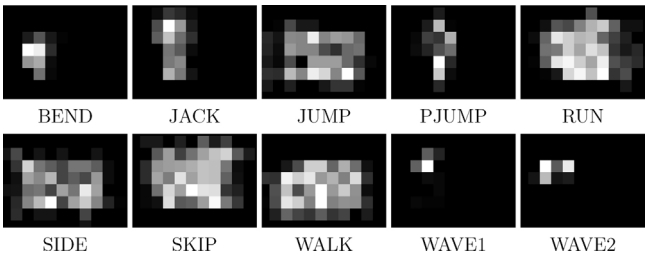

Fig. 2. QP gradient image of the actions on Weizmann dataset [28].

temporally. Specifically, for a k-frame bunch starting from frame t, the QGI (Fig. 2) is defined as:

$$QGI(t,k) = \sum_{j=0}^{k-1} P_{acc}(t+j) \quad (2)$$

QGI located at $[l, m]$ can be represented as $QGI(t, k, l, m)$.

3.1.2. QGI spread feature

It can be observed from the QGIs in Fig. 2 that the horizontal (HS) and vertical spread (VS) of action has clue regarding the same. The VS is more for actions like jack and pump. In case of hand waving actions the spread is localized to the upper torso and hence the VS will be less. Actions wave 1 and wave 2 can be distinguished by HS, even though the VS are comparable. Actions like bend, walk and run have comparable VS. But, bend can be distinguished from run or walk by adding HS too as a feature. The spread feature can distinguish between actions like bend, jack, jump, wave 1 and wave 2. But the confusion between skip, side, run, walk and jump remains as the spreads in both dimension are comparable. Spread is normalized with maximum of frame height or width (in MBs) before adding to the final feature vector.

3.1.3. QGI projection profile feature

The horizontal and vertical QGI projection profiles (Fig. 3) of each action-bunch of k frames starting from frame t, for a video with frame width w and frame height h (in pixels) is given by:

$$QGIP_{horiz}(t,k) = \sum_{j=1}^{w/16} QGI(t,k,l+j,m) \quad (3)$$

$$QGIP_{vert}(t,k) = \sum_{i=1}^{h/16} QGI(t,k,l,m+i) \quad (4)$$

3.1.4. Motion accumulation

The magnitude and orientation of motion vector of a sub-macroblock (4×4 block in H.264/AVC) and location $[l, m]$ in frame t and with horizontal and vertical components u and v respectively are given by:

$$M_t(l,m) = \sqrt[2]{u^2+v^2} \quad (5)$$

$$A_t(l,m) = atan(v/abs(u)) \quad (6)$$

where $atan(x)$ returns the inverse tangent (arctangent) of x in radians. For real elements of x, $atan(x)$ is in the range $[-\pi/2, \pi/2]$. The absolute value of the horizontal component of motion vector is used to attain left–right symmetry of actions. The magnitude of motion vector of each sub-macroblock is accumulated temporally over k frames as given by:

$$MA_t(l,m) = \sum_{j=0}^{k-1} M_{t+j}(l,m) \quad (7)$$

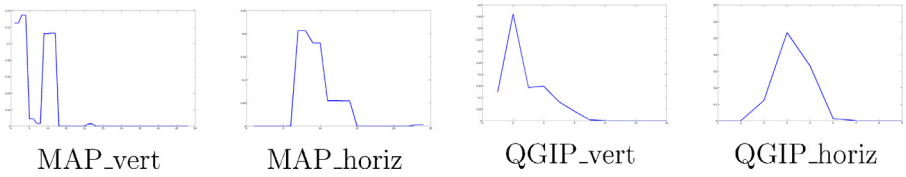

Fig. 3. Projection profiles for the action 'both hands waving (wave 2)'.

The accumulated motion is used to form the horizontal and vertical projection profiles to represent each action.

The motion-accumulated projection profiles are given by:

$$MAP_{horiz}(t,k) = \sum_{j=1}^{w/4} MA_t(l+j, m) \tag{8}$$

$$MAP_{vert}(t,k) = \sum_{i=1}^{h/4} MA_t(l, m+i) \tag{9}$$

Horizontal and vertical profiles, after sub-sampling and normalization, are concatenated to form the projection feature. In the experiments only one in four samples of each MAP profile are used to minimize the feature vector length. The projection feature (Fig. 3) has information on 'where the motion occurred and to what extent'.

3.1.5. Magnitude-weighted-orientation of motion vectors

Each action has a typical set of motion vectors with more frequency of occurrence, which may differ in magnitudes but are similar in orientations. For example, hand-waving actions will have a typical set of repeating MV orientations in the waving-area. To utilize this property, histogram can be used. If raw histograms are employed, actions like running and walking will get confused as the orientation of those actions will be more or less the same. In those cases, the speed of action need to be utilized. The magnitude of the MVs will be more for actions performed at higher speeds. Hence, the histogram of orientations weighted by magnitudes is used as a feature vector, as defined in Eq. (10).

$$MWOM(u) = \sum_{b(A_t(l,m))=u} M_t(l, m) \tag{10}$$

where $MWOM(u)$ is the histogram of the oriented motion vector. $b(A_t(l, m))$ finds the corresponding histogram bin for the angle $A_t(l, m)$.

4. PBL-McRBFN classifier overview

Consider a set of training data samples, where $\mathbf{x}^t \in \Re^m$ is an m-dimensional input of the tth sample, and $c^t \in \{1, \ldots, n\}$ is its class label, and n is the total number of classes. The objective of McRBFN is to approximate the function $\mathbf{x}^t \in \Re^m \rightarrow \mathbf{y}^t \in \Re^n$. McRBFN architecture is same as in [11] and has two components, namely the cognitive component and the meta-cognitive component as shown in Fig. 4. These components are discussed in the following sections.

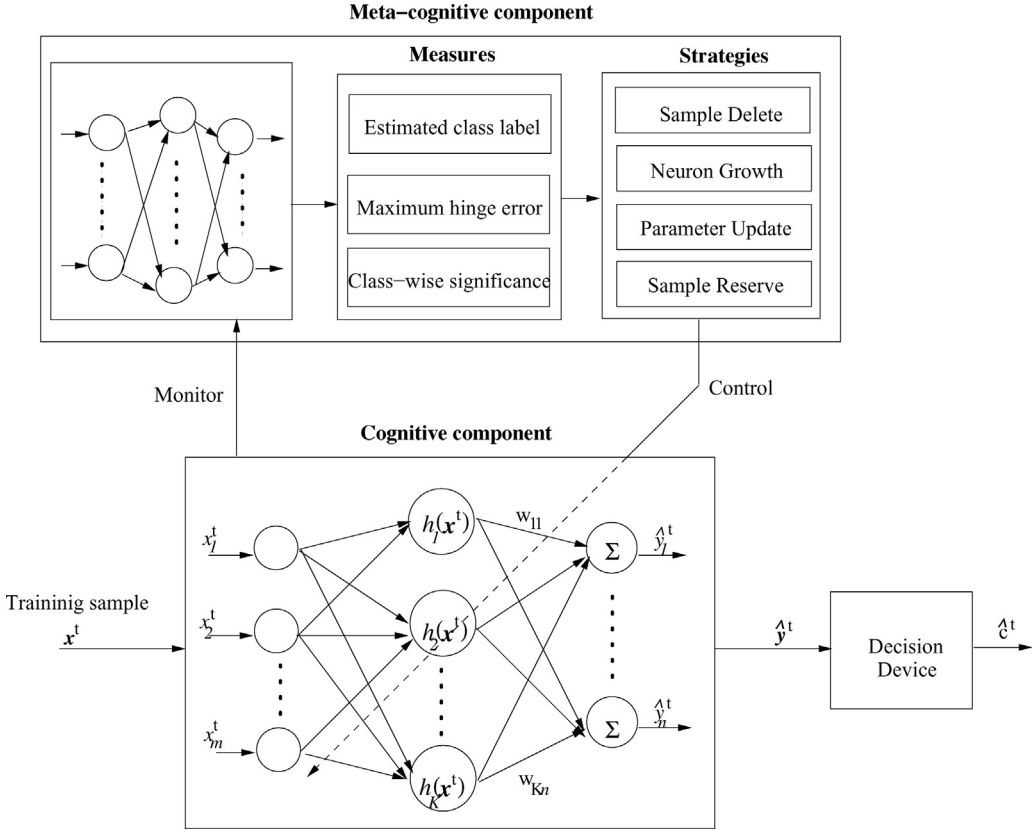

Fig. 4. McRBFN classifier.

4.1. Cognitive component of McRBFN

The cognitive component of McRBFN is a radial basis function network. The hidden layer of the network holds the Gaussian activation function. For a given input $\mathbf{x}^t$, the predicted output $\widehat{y}_j^t$ is given as

$$\widehat{y}_j^t = \sum_{k=1}^{K} w_{kj} h_k^t, \quad j = 1, \ldots, n \tag{11}$$

where w_{kj} is the weight connecting the kth hidden neuron to the jth output neuron and h_k^t is the response of the kth hidden neuron to the input $\mathbf{x}^t$ is given by

$$h_k^t = \exp\left(-\frac{\|\mathbf{x}^t - \boldsymbol{\mu}_k^l\|^2}{(\sigma_k^l)^2} \right) \tag{12}$$

where $\boldsymbol{\mu}_k^l \in \Re^m$ is the center and $\sigma_k^l \in \Re^+$ is the width of the kth hidden neuron. Here, the superscript l represents the corresponding class of the hidden neuron.

Projection based learning algorithm helps to calculate the optimal network output parameters. For t consecutive samples, the error function is

$$J(\mathbf{W}) = \frac{1}{2} \sum_{i=1}^{t} \sum_{j=1}^{n} \begin{cases} 0 & \text{if } y_j^i \widehat{y}_j^i > 1 \\ \left(y_j^i - \widehat{y}_j^i \right)^2 & \text{otherwise} \end{cases} \tag{13}$$

The resultant optimal output weights $\mathbf{W}^*$ corresponding to the error function $J(\mathbf{W}^*)$ is represented as

$$\sum_{k=1}^{K} a_{kp} w_{kj} = b_{pj}, \tag{14}$$

which is a system of linear equation in the form of $\mathbf{W}^* = \mathbf{A}^{-1}\mathbf{B}$. Here, a_{kp} is the projection matrix and b_{pj} is the output matrix. The derivation of this equation can be found in [11].

4.2. Meta-cognitive component of McRBFN

The meta-cognitive component models the dynamics of the cognitive component, its corresponding knowledge measures and the self-regulated thresholds. During the learning process, the meta-cognitive component monitors the cognitive component and updates its dynamic model of the cognitive component. The meta-cognitive component uses predicted class label ($\widehat{c}^t$), maximum hinge loss (E^t), confidence of classifier ($\hat{p}(c^t|\mathbf{x}^t)$) and class-wise significance (ψ_c) as the measure of knowledge in the new training sample. The meta-cognitive component builds two sample-based and two neuron-based learning strategies using the knowledge measures and the self-regulated thresholds. One of these strategies is chosen for the new training sample such that the cognitive component learns them accurately and achieves better generalization performance.

The meta-cognitive component knowledge measures are defined as shown below:

Predicted class label ($\widehat{c}^t$): $\widehat{c}^t = arg \max_{j \in 1,\ldots,n} \widehat{y}_j^t$

Maximum hinge loss (E^t): The hinge loss ($\mathrm{e}^t = [e_1^t, \ldots, e_j^t, \ldots, e_n^t]^T \in \Re^n$ is

$$e_j^t = \begin{cases} 0 & \text{if } y_j^t \widehat{y}_j^t > 1 \\ y_j^t - \widehat{y}_j^t & \text{otherwise} \end{cases} \quad j = 1, \ldots, n \tag{15}$$

The maximum absolute hinge loss (E^t) is given by

$$E^t = \max_{j \in 1,2,\ldots,n} \left| e_j^t \right| \tag{16}$$

Confidence of classifier ($\hat{p}(c^t|\mathbf{x}^t)$) is given as

$$\hat{p}(j|\mathbf{x}^t) = \frac{\min(1, \max(-1, \hat{y}_j^t)) + 1}{2}, \quad j = c^t \tag{17}$$

Class-wise significance (ψ_c): The class-wise distribution portrays as the key impact factor in calculating the performance of the classifier. Let K^c be the number of neurons associated with the class c, then class-wise spherical potential or class-wise significance (ψ_c) is defined as

$$\psi_c = \frac{1}{K^c} \sum_{k=1}^{K^c} h(\mathbf{x}^t, \boldsymbol{\mu}_k^c) \tag{18}$$

The ψ_c is the knowledge contained in the sample, a smaller value of ψ_c (close to 0) indicates that the sample is new.

4.3. Learning strategies

The meta-cognitive component has four learning strategies, which directly addresses the basic principles of self-regulated human learning namely, *what-to-learn*, *when-to-learn* and *how-to-learn* and selects the best strategy for the new training sample.

Sample delete strategy is given by

$$\widehat{c}^t == c^t \,\mathbf{AND}\, \hat{p}(c^t|\mathbf{x}^t) \geq \beta_d \tag{19}$$

where β_d is the deletion threshold. If β_d is close to 1 then all samples participates in the learning which may result in over-training. Please refer [11] for further understanding on deletion thresholds.

Neuron growth strategy is given by

$$(\widehat{c}^t \neq c^t \,\mathbf{OR}\, E^t \geq \beta_a) \,\mathbf{AND}\, \psi_c(\mathbf{x}^t) \leq \beta_c \tag{20}$$

where β_c is the knowledge measurement threshold and β_a is the self-adaptive addition threshold. If β_c is chosen closer to 0, then very few neurons will be added to the network. If β_c is closer to 1, then the resultant network may contain neurons with poor generalization ability. Based on the nearest neuron distances, the new hidden neuron center and width parameters are determined for the different overlapping/no-overlapping conditions as in [11].

When a neuron is added to McRBFN, the output weights are initialized as:

The size of matrix $\mathbf{A}$ is increased from $K \times K$ to $(K+1) \times (K+1)$

$$\mathbf{A}^t = \begin{bmatrix} \mathbf{A}_{K\times K}^{t-1} + (\mathbf{h}^t)^T \mathbf{h}^t & \mathbf{a}_{K+1}^T \\ \mathbf{a}_{K+1} & a_{K+1,K+1} \end{bmatrix} \tag{21}$$

The size of $\mathbf{B}$ is increased from $K \times n$ to $(K+1) \times n$

$$\mathbf{B}_{(K+1)\times n}^t = \begin{bmatrix} \mathbf{B}_{K\times n}^{t-1} + \left(\mathbf{h}^t\right)^T \left(\mathbf{y}^t\right)^T \\ \mathbf{b}_{K+1} \end{bmatrix} \tag{22}$$

and $\mathbf{b}_{K+1} \in \Re^{1\times n}$ is a row vector assigned as

$$b_{K+1,j} = \sum_{i=1}^{K+1} h_{K+1}^i \tilde{y}_j^i, \quad j = 1, \ldots, n \tag{23}$$

After neglecting $\mathbf{h}^t$ vector in Eqs. (21) and (23) the output weights are estimated finally as

$$\begin{bmatrix} \mathbf{W}_K^t \\ \mathbf{w}_{K+1}^t \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{K\times K}^{t-1} & \mathbf{a}_{K+1}^T \\ \mathbf{a}_{K+1} & a_{K+1,K+1} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}_{K\times n}^{t-1} \\ \mathbf{b}_{K+1} \end{bmatrix} \tag{24}$$

where $\mathbf{W}_K^t$ is the output weight matrix for K hidden neurons, and $\mathbf{w}_{K+1}^t$ is the vector of output weights for new hidden neuron after learning from tth sample. Please refer to the neuron growth architecture in [11] to further understand the output weights.

Parameters update strategy: The present (tth) training sample helps to update the output weights of the cognitive component ($\mathbf{W}_K = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K]^T$) if the following condition is satisfied.

$$c^t == \widehat{c}^t \quad \textbf{AND} \quad E^t \geq \beta_u \tag{25}$$

where β_u is the self-adaptive update threshold. If β_u is closer to 50% of E^t, then fewer samples will be used and most of the samples will be dragged to the end of the training sequence. The resultant network will not approximate the function accurately. If a lower value is chosen, then all samples will be used in updating without any change in the training sequence.

The $\mathbf{A} \in \Re^{K\times K}$ and $\mathbf{B} \in \Re^{K\times n}$ matrices updated as

$$\mathbf{A}^t = \mathbf{A}^{t-1} + \left(\mathbf{h}^t\right)^T \mathbf{h}^t; \quad \mathbf{B}^t = \mathbf{B}^{t-1} + \left(\mathbf{h}^t\right)^T \left(\mathbf{y}^t\right)^T \tag{26}$$

Finally the output weights are updated as

$$\mathbf{W}_K^t = \mathbf{W}_K^{t-1} + \left(\mathbf{A}^t\right)^{-1} \left(\mathbf{h}^t\right)^T \left(\mathbf{e}^t\right)^T \tag{27}$$

where $\mathbf{e}^t$ is hinge loss for tth sample as in Eq. (15).

Sample reserve strategy: If the new training sample does not satisfy either the sample deletion or the neuron growth or the cognitive component parameters update criterion, then the current sample is pushed to the rear end of data stream. Since McRBFN modifies the strategies based on the knowledge in the current sample, these samples may be used in later stage. The above process is repeated for every incoming samples.

The self-regulative addition threshold (β_a) ranges from 1.2 to 1.5. When (β_a) is chosen close to 1, then the samples that are misclassified will be used for neuron addition. If chosen close to 2 then very few neurons will be added. The deletion threshold (β_d) ranges from 0.85 to 0.95. If the predicted posterior probability or confidence is greater than (β_d) and predicted class label is same as actual class label then the current sample is similar to existing knowledge in the cognitive component. In that case the current sample is deleted from the training sequence without being used in the learning process. If (β_d) is chosen to be 0.5 then many samples satisfying the condition will be deleted without being used in learning. Hence, the resultant classifier may not provide better generalization performance. If one selects close to 1, say 0.99 then most of the similar samples will be used in learning, which will result in overtraining. The knowledge measurement threshold (β_c) ranges from 0.4 to 0.8. (β_c) helps in identifying the novelty of current sample. It depends on the spherical potential, which is between 0 and 1. If spherical potential is close to 1 then sample is similar to existing knowledge, lesser value of spherical potential indicates that the sample is novel. If one selects the threshold (β_c) close to zero then the network does not allow addition of neurons. Similarly, if one selects the threshold (β_c) close to 1 then all samples will be identified as novel sample. The self-adaptive update threshold (β_u) ranges from 0.3 to 0.8. If (β_u) is chosen close to 1 then no sample will be used in updating and hence the resultant network does not approximate the decision function. If chosen close to 0 then all samples will be used for updating. More details on tuning the thresholds can be found in [10,29].

The action recognition ability of PBL-McRBFN along with the proposed features is evaluated on two standard benchmark datasets in the following sections.

5. Experiments, results and discussion

In this paper, the performance of the proposed system has been evaluated on two benchmark action recognition databases, Weizmann [28] and KTH [30]. In literature, there have been other databases such as UCF101 [31] and HMDB51 [32] proposed. However, these databases involve complex actions being performed by small objects with significant background cluttering. The compressed domain features extracted from these complex databases will be very noisy for classification purpose. So, these databases are not included in the current study.

Table 1
Experimental setup.

Feature extraction	Dataset variants	
	Weizmann	KTH
GOP	(20, 30, 35)	–
Bunch of frames	Yes	Yes
Full video	Yes	Yes

For performance evaluation, group of frames and full video based features are extracted (refer Table 1). In case of full video, a single feature vector is extracted from the entire video which is encoded with GOP 30. The extracted features are input to PBL-McRBFN and the performance is recorded for 10 random trials by leaving two persons for testing. This is further randomized by creating twenty different trials for comparison. Table 1 provides the list of various combinations of datasets that are used in these experiments. It has been shown in these works [33–41] that, for action recognition, BayesNet [42,43], Random Forest [44], SVM [45,46] and ELM [47–51] provide better performance than other algorithms. Hence, in this paper we employ these four algorithms to compare our analysis with PBL-McRBFN. The parameters of these algorithms are optimized as explained in their respective references. Weka tool [52] has been used to implement these algorithms.

5.1. Dataset descriptions

5.1.1. KTH dataset

KTH dataset contains six types of human actions viz., boxing, hand waving, hand clapping, walking, jogging and running. The actions are performed by 25 subjects in four different scenarios: outdoors s1, outdoors with scale variation s2, outdoors with different clothes s3 and indoors s4, as shown in Fig. 5. There are a total of 599 video streams taken over homogeneous backgrounds with 25 fps frame rate and spatial resolution of 160 × 120 pixels.

5.1.2. Weizmann dataset

Weizmann dataset consists of 10 different actions viz., walk, run, jump, gallop sideways, bend, one-hand waving, two-hands waving, jump in place, jumping jack and skip (Fig. 6). The actions are performed by 9 subjects. There are a total of 93 video sequences. Four different variants are analyzed: bunches of 20 frames encoded with GOP sizes 20, 30 and 35, and full video encoded with GOP size

Fig. 5. KTH actions (image courtesy: Laptev et al.[30]).

Fig. 6. Weizmann actions (image courtesy: Liu et al. [53]).

Table 2
KTH database speed results.

Method	Scale	Accuracy	Speed	Domain
Proposed	No	90.44%	**3563** fps	Compressed
Yeo et al. [27]	No	90.00%	–	Compressed
Proposed	Yes	85.00%	**3489** fps	Compressed
Yu et al. [22]	Yes	95.67%	18.98 fps[a]	Pixel
Li et al. [23]	Yes	77.6%	–	Pixel
Sabri et al. [54]	Yes	85.65%	–	Pixel

[a] Implemented in C++ on Intel Core™i7 920 PC.

Table 3
Weizmann dataset classification speed comparison.

Method	Accuracy	Speed	Domain
Proposed	85.56%	**2186** fps	Compressed
Sadek et al. [21]	97.80%	18 fps	Pixel
Rahman et al. [55]	100%	12 fps	Pixel
Jiang et al. [56]	100%	–	Pixel

30. The classification speed achieved is 2186 fps with a standard deviation of 7 frames.

5.2. Comparison of the proposed feature extraction method with state-of-the-art

The comparison results for the two datasets are shown in Tables 2 and 3. For the KTH dataset, some of the previous works limit their implementation on videos which do not have any significant scale change. These are shown as 'No' in the column 'scale' in Table 2. Results for compressed domain approaches are not available for videos containing scale changes. In order to have a fair comparison, we have removed the videos with scale changes when comparing with such results. The results available in pixel domain include these videos with scale changes and hence we have included them while comparing with pixel domain results. It can be seen from the results that even though the classification accuracy of the proposed features is slightly less than other pixel domain approaches, this approach outperforms these approaches in terms of speed by 100–200 times.

Table 4 shows the confusion matrix for the KTH dataset. A major source of error is the confusion between two actions 'Jogging' and 'Running', which are indeed difficult to differentiate for humans too.

Table 4
Per-video confusion matrix on KTH.

Action	Box	hc	hw	Jog	Run	Walk
Boxing	0.90	0.07	0.02	0.00	0.00	0.01
Hand clap	0.08	0.82	0.09	0.00	0.00	0.01
Hand wave	0.06	0.09	0.85	0.00	0.00	0.00
Jogging	0.00	0.00	0.02	0.76	0.13	0.09
Running	0.04	0.00	0.02	0.11	0.82	0.01
Walking	0.00	0.00	0.01	0.04	0.00	0.95

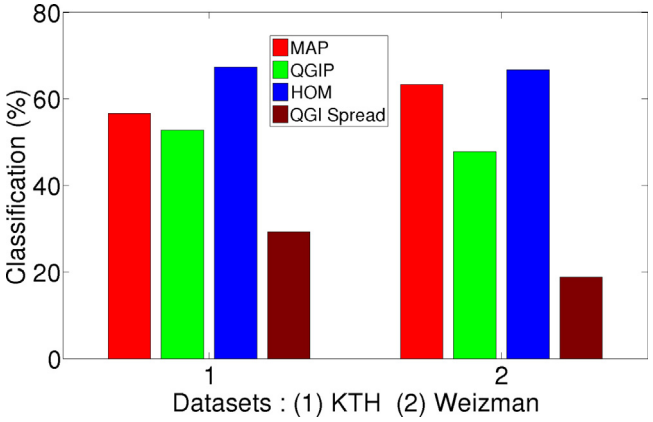

Fig. 7. Feature wise classification performance on KTH and Weizmann datasets.

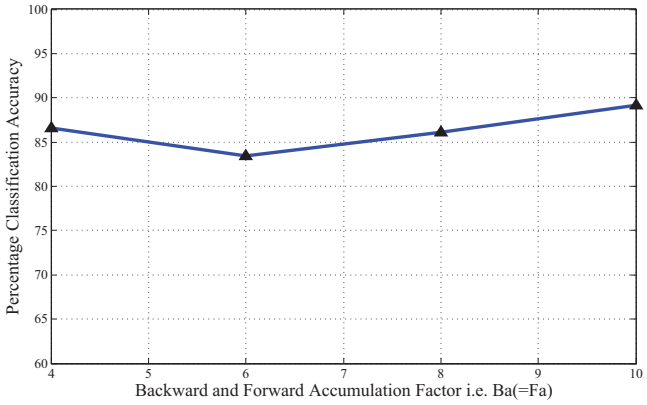

Fig. 8. Effect of accumulation factor on accuracy.

The effect of each individual feature is shown in Fig. 7. It can be seen that none of features is highly dominating to the extent that it can be considered solely responsible for the classification accuracy achieved by the algorithm.

5.3. Effect of parameters on classification accuracy

This section analyzes the sensitivity of the proposed system for various parameters such as accumulation factor and bunch size.

5.3.1. Accumulation factor

The classification accuracy is plotted in Fig. 8 by varying the backward and forward accumulation factors $B_a(=F_a)$ as 4, 6, 8 and 10. The accumulation factors determine how many past and future frames are considered in the computation of the QGI of the current frame. Here, the QGI computed from 21 frames i.e., $B_a = F_a = 10$ is found to be giving relatively better performance.

In order to analyze the effect of unequal values of B_a and F_a, we have tabulated the accuracy values for both the cases i.e. $B_a > F_a$ and $B_a < F_a$. As seen from Table 5, biasing the feature towards past

Table 5
Accuracy for different values of B_a and F_a.

Recgonition accuracy				
	$B_a = 4$	$B_a = 6$	$B_a = 8$	$B_a = 10$
$F_a = 4$	86.95	85.21	82.41	85.74
$F_a = 6$	88.33	83.83	83.63	85.15
$F_a = 8$	87.83	85.50	86.67	85.28
$F_a = 10$	84.72	88.79	87.78	89.58

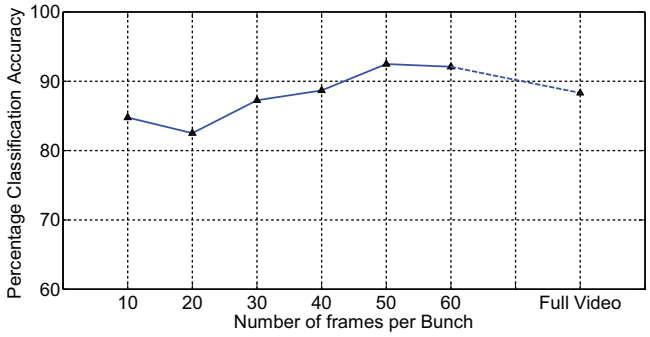

Fig. 9. Effect of bunch size on accuracy.

or future frames does not significantly improve the accuracy of the algorithm.

5.3.2. Bunch size

The effect of bunch size i.e. the number of frames encoded to generate one feature vector is plotted in Fig. 9. The bunch size is varied from 10, 20, 30, 40, 50, 60 and the entire video (full video). It is observed that the accuracy increases as more and more frames are grouped together for feature extraction. This is because now there is more meaningful information in the bunch. When bunch size is small (10–20), each feature vector contains information of only 10–20 frames. This corresponds to less than 1 s in time domain. With increase in the size of each bunch, more information regarding the action flow is included in the feature vector.

This trend does not continue forever. When the entire video is encoded in a single feature vector, the accuracy decreases. This is because now a lot of information is packed in this one single vector. To some extent, it loses its ability to represent the different motions in the action performed.

5.4. Effect of PBL-McRBFN classifier

5.4.1. Weizmann dataset

The results for different GOP sizes (20, 25 and 35) are provided in Table 6. Here, all the video features are extracted for a temporal window (k) of 20 frames. Many real-time surveillance applications require that the system should recognize actions within small time interval, hence a bunch size of 20 frames is chosen which falls well within 1 s time window. The table provides the performance evaluation of PBL-McRBFN with respect to testing maximum accuracy and mean accuracy. The performance is compared against

Table 6
Performance analysis for GOP in Weizmann dataset for bunch of frames $k = 20$.

GOP size	Algorithm	Testing accuracy		
		Max	Mean	S.D.
20	BayesNet	91.3	81.37	0.0993
	RandomForest	86.95	80.12	0.0683
	SVM	82.61	73.91	0.87
	ELM	82.608	77.17	0.036
	PBL-McRBFN	91.3	90.06	0.0124
25	BayesNet	89.47	82.45	0.0702
	RandomForest	89.47	77.04	0.1243
	SVM	78.94	76.6	0.0234
	ELM	86.95	80.97	0.0372
	PBL-McRBFN	94.74	90.06	0.0468
35	BayesNet	82.61	73.91	0.087
	RandomForest	91.3	80	0.113
	SVM	85.96	75.45	0.1051
	ELM	91.30	82.60	0.05325
	PBL-McRBFN	95.65	91.3	0.0435

Table 7
Confusion matrix of Weizmann for GOP size = 35.

PBL-McRBFN	a	b	c	d	e	f	g	h	i	j
a	3	0	0	0	0	0	0	0	0	0
b	0	3	0	0	0	0	0	0	0	0
c	0	0	2	0	0	0	0	0	0	0
d	0	0	0	2	0	0	0	0	0	0
e	0	0	0	0	2	0	0	0	0	0
f	0	0	0	0	0	1	0	1	0	0
g	0	0	0	0	0	0	2	0	0	0
h	0	0	0	0	0	0	0	2	0	0
i	0	0	0	0	0	0	0	0	2	0
j	0	0	0	0	0	0	0	0	0	3

Table 8
Performance analysis for full video in Weizmann dataset with GOP size 30.

Algorithm	Testing accuracy		
	Max	Mean	S.D.
BayesNet	86.95	80	0.0695
RandomForest	91.3	86.95	0.0435
SVM	91.3	76.08	0.1522
ELM	91.3	81.15	0.0502
PBL-McRBFN	95.65	91.73	0.0392

BayesNet, RandomForest standard SVM and ELM classifiers. From the table, it could be observed that PBL-McRBFN performs significantly better when compared to other algorithms. In particular, the following can be observed:

- PBL-McRBFN and BayesNet achieves 91% for GOP-20. In random validation, mean accuracy for BayesNet drops by 10% whereas PBL-McRBFN drops by 1%. PBL-McRBFN performs atleast 10% better than RandomForest, SVM and ELM classifiers.
- For GOP-25, PBL-McRBFN achieves 94%. In random validation, mean accuracy for RandomForest drops by 12% whereas PBL-McRBFN drops by 4%. PBL-McRBFN performs atleast 13% better than RandomForest and, SVM and ELM classifiers.
- For GOP-35, PBL-McRBFN achieves 95%. In random validation, mean accuracy for RandomForest and SVM drops by 10% whereas PBL-McRBFN drops by 4%. PBL-McRBFN performs atleast 16% better than BayesNet, SVM and ELM classifiers.

The average performance of PBL-McRBFN remains constant across various values of GOP. A sample confusion matrix is presented for GOP-35 in Table 7, for PBL-McRBFN classifier.

Table 8 clearly shows that PBL-McRBFN achieves 95% accuracy in full video. Random validation drops the accuracy of SVM by 15% whereas PBL-McRBFN drops by 4%. PBL-McRBFN performs atleast 10% better than BayesNet SVM and ELM classifiers.

5.4.2. KTH dataset

Table 9 clearly shows that PBL-McRBFN achieves 90% accuracy in video taken as a bunch of 100 frames with fixed GOP size of 30.

Table 9
Performance analysis for KTH dataset with 100 frames time window.

Algorithm	**Testing accuracy**		
	Max	Mean	S.D.
BayesNet	82.03	74.77	0.0726
RandomForest	86.01	83.57	0.0244
SVM	88.75	82.58	0.0617
ELM	89.33	81.39	0.0537
PBL-McRBFN	90.4	85.11	0.0423

Table 10
Performance analysis for full video in KTH dataset.

Algorithm	Testing accuracy		
	Max	Mean	S.D.
BayesNet	85	78.06	0.0694
RandomForest	90	85.83	0.0417
SVM	85	80.99	0.0401
ELM	88.47	82.6	0.0395
PBL-McRBFN	96	88.78	0.0789

Random validation drops the accuracy of BayesNet by 8% whereas PBL-McRBFN drops by 5%. PBL-McRBFN performs atleast 10% better than BayesNet and 3% better than SVM and ELM classifiers.

Table 10 clearly shows that PBL-McRBFN achieves 89% accuracy in full video. Random validation shows drop in the accuracy of RandomForest by 3%. PBL-McRBFN performs 8% better than BayesNet, SVM and ELM classifiers.

From the experiments carried out, it is evident that PBL-McRBFN performs better in comparison to other state-of-the-art algorithms. The meta-cognitive learning strategies employed in McRBFN has helped the network avoid over-training and as a result, attain better generalization.

6. Conclusion and future work

In this paper, the performance of QGI based compressed domain features accumulated over group of frames and full video with meta-cognitive learning classifier for action recognition in the compressed domain is demonstrated. Here, the features extracted are QGI Spread, QGIP and MAP for evaluation. The classification is performed with PBL-McRBFN, which shows superior recognition accuracy. The performance is evaluated using 2 benchmark data sets and the recognition performance is compared with BayesNet, RandomForest, SVM and ELM. PBL-McRBFN employs sequential learning with meta-cognition and hence approximates the decision surface more efficiently by proper selection of samples in learning. From the results, it is observed that PBL-McRBFN classifier outperforms other classifiers by more than 10%. The proposed approach can classify QCIF resolution videos with speed more than 2000 fps and is more than 100 times faster than existing state-of-the-art pixel domain approaches. The proposed approach may not handle occluded actions and more than one action in a given video accurately. One needs to modify the existing feature and adaptive classifier to handle these situations.

Acknowledgement

The authors would like to thank the reviewers for their valuable comments which has improved the quality of the paper.

References

[1] R. Babu, M. Tom, P. Wadekar, A survey on compressed domain video analysis techniques, Multimed. Tools Appl. (2014) 1–36, http://dx.doi.org/10.1007/s11042-014-2345-z
[2] T. Wiegand, G. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC video coding standard, IEEE Trans. Circuits Syst. Video Technol. 3 (7) (2003) 560–576.
[3] S. Biswas, R.G. Praveen, R.V. Babu, Super-pixel based crowd flow segmentation in H.264 compressed videos, in: International Conference on Image Processing, 2014.
[4] T.-O. Nelson, L. Narens, Metamemory: a theoretical framework and new findings, Psychol. Learn. Mot. 26 (C) (1990) 125–173.
[5] K. Subramanian, S. Suresh, N. Sundararajan, A metacognitive neuro-fuzzy inference system (McFIS) for sequential classification problems, IEEE Trans. Fuzzy Syst. 21 (6) (2013) 1080–1095.
[6] K. Subramanian, S. Suresh, Human action recognition using meta-cognitive neuro-fuzzy inference system, Int. J. Neural Syst. 22 (6) (2012), 1250028 (15).
[7] K. Subramanian, S. Suresh, A meta-cognitive sequential learning algorithm for neuro-fuzzy inference system, Appl. Soft Comput. 12 (2012) 3603–3614.
[8] R. Savitha, S. Suresh, N. Sundararajan, Metacognitive learning in a fully complex valued radial basis function neural network, Neural Comput. 24 (5) (2012) 1297–1328.
[9] R. Savitha, S. Suresh, N. Sundararajan, A meta-cognitive learning algorithm for a fully complex-valued relaxation network, Neural Netw. 32 (Special Issue) (2012) 209–218.
[10] G. Sateesh Babu, S. Suresh, Meta-cognitive neural network for classification problems in a sequential learning framework, Neurocomputing 81 (1) (2012) 86–96.
[11] G. Babu, S. Suresh, Meta-cognitive RBF network and its projection based learning algorithm for classification problems, Appl. Soft Comput. 13 (1) (2013) 654–666.
[12] G. Babu, S. Suresh, Sequential projection based metacognitive learning in a radial basis function network for classification problems, IEEE Trans. Neural Netw. Learn. Syst. 24 (2) (2013) 194–206.
[13] S. Suresh, K. Subramanian, A sequential learning algorithm for meta-cognitive neuro-fuzzy inference system for classification problems, in: The 2011 International Joint Conference on Neural Networks (IJCNN), 2011, pp. 2507–2512.
[14] K. Subramanian, S. Suresh, A meta-cognitive sequential learning algorithm for neuro-fuzzy inference system, Appl. Soft Comput. 12 (11) (2012) 3603–3614.
[15] R. Savitha, S. Suresh, H. Kim, A meta-cognitive learning algorithm for an extreme learning machine classifier, Cognit. Comput. 6 (2) (2014) 253–263.
[16] C.-A. Lin, Y.-Y. Lin, H.-Y.M. Liao, S.-K. Jeng, Action recognition using instance-specific and class-consistent cues, in: International Conference on Image Processing, 2012.
[17] S.M. Amiri, P. Nasiopoulos, V.C. Leung, Non-negative sparse coding for human action recognition, in: International Conference on Image Processing, 2012.
[18] B. Wu, C. Yuan, W. Hu, Human action recognition based on a heat kernel structural descriptor, in: International Conference on Image Processing, 2012.
[19] X. Zhang, Z. Miao, L. Wan, Human action categories using motion descriptors, in: International Conference on Image Processing, 2012.
[20] Y. Wang, G. Mori, Max-margin hidden conditional random fields for human action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 872–879.
[21] B.M.S. Sadek, A. Al-Hamadi, U. Sayed, A fast statistical approach for human activity recognition, International Journal of Intelligence Science 2 (1) (2012) 9–15.
[22] T.-H. Yu, T.-K. Kim, R. Cipolla, Real-time action recognition by spatiotemporal semantic and structural forests, in: BMVC, 2010, pp. 1–12.
[23] Z. Li, Y. Fu, T. Huang, S. Yan, Real-time human action recognition by luminance field trajectory analysis, in: Proceedings of the 16th ACM International Conference on Multimedia, 2008, pp. 671–676.
[24] B. Ozer, W. Wolf, A. Akansu, Human activity detection in mpeg sequences, in: Workshop on Human Motion, Proceedings, 2000, pp. 61–66.
[25] R.V. Babu, B. Anantharaman, K.R. Ramakrishnan, S.H. Srinivasan, Compressed domain action classification using HMM, Pattern Recognit. Lett. 23 (2002) 1203–1213.
[26] R.V. Babu, K.R. Ramakrishnan, Recognition of human actions using motion history information extracted from the compressed video, Image Vision Comput. 2 (8) (2004) 597–607.
[27] C. Yeo, P. Ahammad, K. Ramchandran, S. Sastry, High-speed action recognition and localization in compressed domain videos, IEEE Trans. Circuits Syst. Video Technol. 8 (8) (2008) 1006–1015.
[28] M. Blank, L. Gorelick, E. Shechtman, M. Irani, R. Basri, Actions as space-time shapes, in: The Tenth IEEE International Conference on Computer Vision, 2005, pp. 1395–1402.
[29] G. Babu, R. Savitha, S. Suresh, A projection based learning in meta-cognitive radial basis function network for classification problems, in: The 2012 International Joint Conference on Neural Networks IJCNN, 2012, pp. 1–8.
[30] C. Schldt, I. Laptev, B. Caputo, Recognizing human actions: a local SVM approach, in: In Proc. ICPR, 2004, pp. 32–36.
[31] K. Soomro, A. R. Zamir, M. Shah, Ucf101. A dataset of 101 human actions classes from videos in the wild, CoRR abs/1212.0402.
[32] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, HMDB: a large video database for human motion recognition, in: Proceedings of the International Conference on Computer Vision (ICCV), 2011.
[33] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local SVM approach, in: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004, vol. 3, IEEE, 2004, pp. 32–36.
[34] R. Poppe, A survey on vision-based human action recognition, Image Vision Comput. 28 (6) (2010) 976–990.
[35] H. Jhuang, T. Serre, L. Wolf, T. Poggio, A biologically inspired system for action recognition, in: IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007, 2007, pp. 1–8.
[36] N. Robertson, I. Reid, A general method for human activity recognition in video, Comput. Vision Image Underst. 104 (2006) 232–248 (special issue on modeling people: vision-based understanding of a person's shape, appearance, movement and behaviour.
[37] D. Patterson, D. Fox, H. Kautz, M. Philipose, Fine-grained activity recognition by aggregating abstract object usage, in: Ninth IEEE International Symposium on Wearable Computers, 2005, pp. 44–51.
[38] J. Gall, A. Yao, N. Razavi, L. Van Gool, V. Lempitsky, Hough forests for object detection, tracking, and action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2188–2202.

[39] A. Yao, J. Gall, L. Van Gool, A hough transform-based voting framework for action recognition, in: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 2061–2068.
[40] R. Minhas, A. Baradarani, S. Seifzadeh, Q.J. Wu, Human action recognition using extreme learning machine based on visual vocabularies, Neurocomputing 73 (2010) 1906–1917 (subspace learning /selected papers from the European Symposium on Time Series Prediction).
[41] A. Mohammed, R. Minhas, Q.J. Wu, M. Sid-Ahmed, Human face recognition based on multidimensional {PCA} and extreme learning machine, Pattern Recognit. 44 (2011) 2588–2597 (semi-supervised learning for visual content analysis and understanding).
[42] G. Cooper, E. Herskovits, A Bayesian Method for Constructing Bayesian Belief Networks from Databases, 1990, pp. 86–94.
[43] G. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Mach. Learn. 9 (4) (1992) 309–347.
[44] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[45] C.-C. Chang, C.-J. Lin, LIBSVM – A Library for Support Vector Machines, the Weka Classifier Works with Version 2.82 of LIBSVM, 2001.
[46] Y. EL-Manzalawy, WLSVM, 2005, http://www.cs.iastate.edu/ỹasser/wlsvm/.
[47] G.-B. Huang, An insight into extreme learning machines: random neurons, random features and kernels, Cognit. Comput. (2014) 1–15.
[48] R. Savitha, S. Suresh, H. Kim, A meta-cognitive learning algorithm for an extreme learning machine classifier, Cognit. Comput. 6 (2014) 253–263.
[49] R. Venkatesh Babu, B. Rangarajan, Human action recognition in compressed domain using PBL-McRBFN approach, in: IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014, pp. 1–6.
[50] R. Savitha, S. Suresh, N. Sundararajan, Fast learning circular complex-valued extreme learning machine (CC-ELM) for real-valued classification problems, Inform. Sci. 187 (2012) 277–290.
[51] S. Suresh, S. Saraswathi, N. Sundararajan, Performance enhancement of extreme learning machine for multi-category sparse data classification problems, Eng. Appl. Artif. Intell. 23 (7) (2010) 1149–1157.
[52] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[53] C. Liu, P.C. Yuen, Human action recognition using boosted eigen actions, Image Vision Comput. 28 (5) (2010) 825–835.
[54] A. Sabri, J. Boonaert, S. Lecoeuche, E. Mouaddib, Human action classification using SURF based spatio-temporal correlated descriptors, in: 2012 19th IEEE International Conference on Image Processing (ICIP), 2012, pp. 1401–1404.
[55] S. Rahman, S.-Y. Cho, M. Leung, Recognising human actions by analysing negative spaces, Comput. Vis. IET 6 (3) (2012) 197–213.
[56] Z. Jiang, Z. Lin, L. Davis, Recognizing human actions by learning and matching shape-motion prototype trees, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 533–547.