Human Action Recognition using Meta-Cognitive Neuro-Fuzzy Inference System

K. Subramanian
School of Computer Engineering, Nanyang Technological University
Nanyang Avenue, Singapore 639798
E-mail: kartick1@e.ntu.edu.sg

S. Suresh
School of Computer Engineering, Nanyang Technological University
Nanyang Avenue, Singapore 639798
Corresponding Author: E-mail: ssundaram@ntu.edu.sg

Received (to be inserted
Revised by Publisher)

We propose a sequential Meta-Cognitive learning algorithm for Neuro-Fuzzy Inference System (McFIS) to efficiently recognize human actions from video sequence. Optical flow information between two consecutive image planes can represent actions hierarchically from local pixel level to global object level, and hence are used to describe the human action in McFIS classifier. McFIS classifier and its sequential learning algorithm is developed based on the principles of self-regulation observed in human meta-cognition. McFIS decides on what-to-learn, when-to-learn and how-to-learn based on the knowledge stored in the classifier and the information contained in the new training samples. The sequential learning algorithm of McFIS is controlled and monitored by the meta-cognitive components which uses class-specific, knowledge based criteria along with self-regulatory thresholds to decide on one of the following strategies: a) sample deletion b) sample learning and c) sample reserve. Performance of proposed McFIS based human action recognition system is evaluated using benchmark Weizmann and KTH video sequences. The simulation results are compared with well known SVM classifier and also with state-of-the-art action recognition results reported in the literature. The results clearly indicates McFIS action recognition system achieves better performances with minimal computational effort.

Keywords: Meta-cognition, neuro-fuzzy inference system, action recognition, Weizmann, KTH.

1. Introduction

Recognizing human action is an important aspect in various machine vision applications such as, video analytics, gesture recognition, video indexing, and man-machine interface. It is, in its own respect, one of the most promising, yet immensely difficult task. The challenge in development of human action recognition system is two fold, viz, (a) extraction of information rich features (action representation) and (b) efficient mapping of features to each action (action recognition). The features and the algorithm chosen for the recognition of human action should be able to overcome the problems arising due to self-occlusion, cluttering, viewpoint change, intra-class variations, etc. Although numerous research works are available which try to tackle the above problems, few works are available which employ both local pixel level as well as global object level information, which are required for effective modeling and representation of action. It has been reported that optical flow vectors can extract simple dynamic features hierar-
Human Action Recognition using McFIS

Technically from local pixel level to global object level. Therefore, in this paper, we represent motion features corresponding to each action by extracting the accumulated motion information (optical-flow) over a small time window.

The aim of any action recognition system is to find the functional relationship between the feature vectors and the action classes. Machine learning algorithms are bestowed with the ability to learn the functional relationship between the inputs and the output, from examples. Since the video for action recognition almost always arrive sequentially and are massive in size, we require algorithms which can learn from stream of data in an online fashion. Artificial neural networks with their excellent learning algorithms such as and fuzzy inference systems with their knowledge representation ability are two commonly employed machine learning techniques for such problems.

In recent times, another type of networks known as neuro-fuzzy inference systems which combines the predictability and interpretability characteristics of neural networks and fuzzy inference systems are gaining popularity in research community. In recent times, these neuro-fuzzy inference systems have been employed for solving problems in various domains such as load forecasting, system modeling, industrial control, etc. A few works can also be found wherein neuro-fuzzy systems are employed for human activity prediction. Works are also available in literature which combine properties of wavelets into the learning of fuzzy networks.

One of the first neuro-fuzzy network based on adaptive fuzzy systems was by Kasabov and is based on the principle of resource allocation network where the rules and parameters are evolved through incremental hybrid supervised/unsupervised online learning. Dynamic Evolving Fuzzy Inference System (DENFIS) uses evolving clustering method to add rules. While self constructing neuro-fuzzy inference network and dynamic fuzzy neural network uses self organized rule generation, the former uses recursive least square to adapt the parameters. Evolving Takagi Sugeno and its variant SimplTS has been proposed for learning Takagi Sugeno Kang (TSK) type fuzzy systems by updating the rules and parameters in supervised and unsupervised fashion. Sequential Adaptive Fuzzy Inference System (SAFIS) is the first truly sequential algorithm which uses statistical measure of significance to add and prune rules. Extended version of SAFIS (ESAIFS) uses, modified influence of a fuzzy rule for adding/pruning and recursive least square error is used to adjust the parameter of the network. None of the algorithms available in neuro-fuzzy hybrid domain practice self-regulatory learning.

Recent work on human learning ability indicates that the learning is effective if one practices a self-regulatory learning approach. The self-regulatory learning is an integral part of human learning, termed as meta-cognition. Meta-cognition refers to the ability of a human brain to determine what-to-learn, when-to-learn and how-to-learn i.e., ability to identify specific piece of knowledge, judge when to start/stop learning by employing a best learning strategy. In literature it has been shown that meta-cognitive learning, which provides a learner with self-regulation, is the best learning strategy. A few works are available in the arena of neural networks literature which employ meta-cognitive strategies for training a network. These networks have been shown to possess efficient learning ability. These works have been extended to complex domain.

Any neuro-fuzzy inference system employing the concept of meta-cognition can be termed as a Meta-Cognitive Neuro-Fuzzy Inference System (McFIS). In this paper, we employ such a meta-cognitive neuro fuzzy inference system to efficiently learn the underlying decision boundary in action recognition problem. In order to validate the performance of McFIS for learning optical-flow features, we perform a comparative study with Support Vector Machine (SVM) and other state-of-the-art methods. These methods will be validated for Weizmann data set and KTH data set. For Weizmann data set, about 70% of the data will be employed for training and rest for testing, whereas for KTH data set, about 50% of data will be employed for training. Optical flow features are extracted from the given data set for recognition and are employed with McFIS and SVM. The results indicate better classification ability of McFIS classifier. The advantage of optical flow features with proposed McFIS classifier are clearly illustrated by comparing the performance with other approached reported in literature.

This paper is organized as follows: In the next
section the background and related works regarding
the action representation and recognition will be pro-
vided. In section 3, details regarding the extraction
of optical flow information is provided. The meta-
cognitive sequential learning algorithm for neuro-
fuzzy inference system based classifier is provided in
section 4. Section 5 provides the performance eval-
uation of McFIS on various action recognition prob-
lems. The paper is concluded in section 6.

2. Background and Related Work

In most of the feature representation techniques,
one of the first steps involved is foreground detection
41. Earlier approaches for action representation in-
volved image based representation, where the region
of interest obtained upon background subtraction
were employed for feature extraction. Such a tech-
nique was employed by Goldenberg et. al42, where
Eigen shapes of foreground silhouettes is extracted
for representation. The drawback of such methods
is their lack of information regarding the motion of
the silhouettes. Polana and Nelson43 and Seitz and
Dyer44, tap in techniques based on the periodicity
analysis. But these techniques are limited to recog-
nition of cyclic actions.

In recent times, research works have concentrated
on analyzing actions in a video sequence by consider-
ing them as space-time volumes45,46 of gradients,
intensities, etc. In addition, some of the recent works
in the field of object recognition have demonstrated
that silhouettes contain detailed information about
the shape of objects47,39. In their initial work, Bob-
lick and Davis48 extracted silhouettes from video se-
quence and constructed binary motion energy image
which indicates where motion occurs. Motion his-
tory images are also constructed, where the pixel in-
tensities are recency function of the silhouette mo-
tion. The two templates are compared using Hu
moments49. These space-time volume over silhou-
ette images have been proven to have the capability
to handle phase correlation. Gorelick et. al39 stacked
silhouettes over a given sequence to form space-time
volume. Then the solution of the Poisson equation
is used to derive local space-time saliency and orien-
tation features. Global features for a given temporal
range are obtained by calculating weighted moment
over these local features. These features are then
used to recognize actions using nearest neighbor clas-
sifiers. In literature50, Histogram of Oriented Gradi-
ents (HOG)51 has also been employed to focus on
foreground edges by non-negative matrix factoriza-
tion. The high dimensionality of HOG features are
reduced using principal component analysis52.

Authors have also extracted feature vectors from
segments of silhouette tunnels53 where, the clas-
sification is performed by approximating the log-
covariance of query segment by sparse linear com-
bination of action dictionaries. Efros et al.4 used
blurred optical flows to recognize actions of players in
football game occupying a very small region in image
plane. The motion descriptors are constructed from
the positive and negative components of horizontal
and vertical flow vectors.

In contrast to global representation, local repre-
sentation provide better robustness to viewpoint
variation, occlusion and noisy background subtrac-
tion. In local representation, the video is encoded
as a collection of local patches. Shao et al.54 rep-
resented action using correlogram of body poses ob-
tained from the silhouette sequences. In order to
avoid the issues of viewpoint variations and partial
occlusion the region of interest is divided into fixed
grids, where each grid describe the image observa-
tion locally. Another commonly used local descrip-
tor is 3-dimensional Harris corner detector46, where
the original Harris corner detector is extended to 3-
dimension. More details on previous works on action
representation and recognition could be availed from
the reviews2,3.

Most of the widely used techniques rely on
computationally expensive features, extracted from
space-time volumes or silhouette tunnels or optical
flow information, which are not suitable for real-time
applications. Optical flow information between con-
secutive image frames provide vital cue about the
object dynamics. In literature, many works could
be found which employ either local features such as
space-time interest points or global features like sil-
houettes. But for effective modeling of the action,
both local and global level dynamics are required.
It has been reported that optical flow vectors can
extract simple dynamic features hierarchically from
local pixel level to global object level. Therefore,
in this paper, we represent motion features corre-
sponding to each action by extracting the accumu-
lated motion information (optical-flow) over a small
time window.
Next section provides the details of optical flow feature extraction from frame level and subsequence level.

3. Optical Flow Feature Extraction from Video

One of the most challenging tasks in action recognition is action representation. In many real-life scenarios, background clutter, occlusion/self-occlusion, scale variations, etc., make action recognition a difficult task. Many works are available which aim to find the best possible representation of action features. The motion information between two frames is a vital source of information to model human actions. In this paper, motion flow information between consecutive frames are employed for action representation.

3.1. Action Representation

In this paper, motion flow between consecutive frames are obtained by employing Lukas and Kanade’s optical flow method. Action is represented here as accumulated motion flow over a brief period of time window (4-6 frames) as given by

$$G_k(i,j) = \sum_n F_n(i,j), \ n \in \{k-w, \cdots, k\}$$  \hspace{1cm} (1)

where, $G_k(i,j)$ is the accumulated flow for the $k^{th}$ frame at $(i,j)^{th}$ location and $F_n(i,j)$ is the optical flow between $n^{th}$ and $(n-1)^{th}$ frame at $(i,j)^{th}$ location.

3.2. Feature Extraction

The optical flow for each frame of video is obtained initially, as shown in section 3.1. The flow vectors of the current frame are accumulated over past few frames (six frames in this study). The object center for each frame is obtained from the mask generated by subtracting the given background video frame from the current frame. In this experiment we employ fifty four rectangular patches of hierarchically increasing sizes, about the object center. The mean value of non-zero motion vectors in each patch is computed as the representative motion vector, given by

$$F_b^k = \frac{\sum_{(i,j) \in R_b} G_k(i,j)}{\sum_{(i,j) \in R_b} I_k(i,j)}$$  \hspace{1cm} (2)

where,

$$I_k(i,j) = \begin{cases} 1 & \text{if } |G_k(i,j)| > 0, \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (3)

Here, $F_b^k$ indicates the representative motion vector corresponding to $b^{th}$ rectangular block in $k^{th}$ frame, $R_b$ indicates the pixels that belong to $b^{th}$ rectangular block and $I(i,j)$ is a binary matrix indicating the locations of non-zero motion vectors. The final frame-level features are extracted by cascading the representative motion vector of each block as,

$$x^f_k = [F_1^k \ F_2^k \ \cdots \ F_{54}^k]$$  \hspace{1cm} (4)

The process of frame level feature extraction is depicted in fig. 2. The feature vectors are derived by finding the mean optical flow within each of the
rectangular patches surrounding the object center, in an hierarchical process. At first, the mean flow vectors are obtained at finer resolution using thirty six windows of size six-by-four, symmetrically placed around the image center (ref. Fig. 2(a)). The remaining features are obtained by merging the windows to sizes of 12×8 (Fig. 2(b)), 18×12 (Fig. 2(c)), 36×12 (Fig. 2(d)), 18×24 (Fig. 2(e)) and 36×24 (Fig. 2(f)).

The final representative motion vectors of all the 54 patches are collected as feature vector. Since each of the motion vector has x and y components, there are totally 108 dimensions.

For recognizing actions in sub-sequence level, the video is segmented into many overlapping sub shots. In this paper, each sub shot contains 8 consecutive frames. The feature vector for each sub shot is obtained by combining individual frame features as

\[ \mathbf{x}_k^l = [\mathbf{x}_{k-\tau}^l \cdots \mathbf{x}_k^l] \quad (5) \]

where, \( \mathbf{x}_k^l \) is the feature vector for \( k^{th} \) sub-sequence and \( \mathbf{x}_k^l \) is the feature vector of the \( k^{th} \) frame. The size of each \( k^{th} \) sub-sequence is 108×8.

In this experiment, all the feature vectors are normalized such that they are in the range [-1,1]. The objective of the work is approximate the functional relationship between the local/global optical flow features and action label. In the next section, we shall define the problem to be solved and propose a neuro-fuzzy inference system classifier which learns based on human meta-cognition, to solve the action recognition problem.

4. Meta-Cognitive Neuro-Fuzzy Inference System

In this paper, human action recognition is considered a supervised classification problem where the aim is to recognize a query video segment using knowledge gained from previously trained video segments. As the data may arrive real-time, it is impractical to store the learned samples and retrain the network. Therefore it is necessary to efficiently learn the functional relation between different action representation to their corresponding action class, as the sample is presented to the network in a single pass. Hence in this paper, we propose a meta-cognitive sequential learning algorithm for a neuro-fuzzy inference system based classifier for efficiently learning the functional relationship between input and output in order to efficiently classify the decision boundary.

4.1. Problem Definition

Let us assume that we have a stream of training data \( \{ (\mathbf{x}^1, c^1), \cdots, (\mathbf{x}^t, c^t), \cdots \} \), where \( \mathbf{x}^t = [x_1^t, \cdots, x_m^t]^T \in \mathbb{R}^m \) is a \( m \)-dimensional input vector of \( t^{th} \) sample, \( c^t \in [1, 2, \cdots, n] \) is the corresponding class label, and \( n \) represents the total number of distinct actions present in the data set. For solving the classification problem in a neuro-fuzzy framework, the class label \( (c^t) \) is converted into a coded class label \( (\mathbf{y}^t = [y_1^t, \cdots, y_j^t, \cdots, y_n^t] \in \mathbb{R}^n) \) as shown below

\[ y_j^t = \begin{cases} 1 & \text{If } j = c^t \\ -1 & \text{otherwise} \end{cases} \quad j = 1, 2, \cdots, n \quad (6) \]

The input space \( \mathbf{x} \) provides the necessary information on the probability distribution over the observation data to predict the corresponding class la-
inference system as shown in Fig. 3.

In this paper we employ a four layer neuro-fuzzy inference system as shown in Fig. 3.

The predicted output ($\hat{y}^t$) using McFIS is given by

$$\hat{y}^t = f(x^t, w)$$  \hspace{1cm} (7)

where the vector $w$ are the parameters of McFIS classifier.

4.2. McFIS Classifier

Meta-cognitive neuro-fuzzy inference system consists of two basic components, a TSK Type-0 neuro-fuzzy inference system which forms the cognitive component and a meta-cognitive component which monitors the cognitive components and controls it.

4.2.1. Neuro-Fuzzy Inference System

In this paper we employ a four layer neuro-fuzzy inference system as shown in Fig. 3.

Input Layer consists of $m$ nodes, each representing one input feature. The input from this node is transmitted directly to the Gaussian rule layer. The output of $i$th input node is

$$u_i = x_i \quad i = 1, 2, \cdots, m$$  \hspace{1cm} (8)

Gaussian Layer consists of $K$ rules, where, each rule computes the overall contribution of the membership function values of the input features. The firing strength $\phi_i$ is given by

$$\phi_i(u) = \exp \left( -\frac{|u - \mu_i|^2}{2\sigma_i^2} \right), \quad i = 1, 2, \cdots, K$$  \hspace{1cm} (9)

where $\mu_i$ is the center of the $i$th Gaussian node and $\sigma_i$ is the width of the $i$th node. Here, $l$ indicates the associated class label of the rule.

Normalization Layer contains as many nodes as Gaussian layer. The output of $i$th normalized node is given by

$$\bar{\phi}_i = \frac{\phi_i}{\sum_{k=1}^{K} \phi_k}, \quad i = 1, 2, \cdots, K$$  \hspace{1cm} (10)

Output Layer: The number of nodes in this layer is equal to the number of distinct classes ($n$). The predicted output ($\hat{y}^t$) is the weighted sum of the normalized output and is given by

$$\hat{y}^t_j = \sum_{k=1}^{K} \alpha_{jk} \bar{\phi}_k, \quad j = 1, 2, \cdots, n$$  \hspace{1cm} (11)

where $\alpha_{jk}$ is the weight connecting the $k$th normalized node and the $j$th output node.

The objective of the classifier is to minimize the error between the predicted output ($\hat{y}^t$) and the actual output ($y^t$). In McFIS, we use the hinge loss error ($e = [e_1, \cdots, e_n]^T \in \mathbb{R}^n$) given by

$$e_j = \begin{cases} 0 & \text{if } \hat{y}^t_j y^t_j > 1 \\ \hat{y}^t_j - y^t_j & \text{otherwise} \end{cases} \quad j = 1, 2, \cdots, n$$  \hspace{1cm} (12)

The maximum absolute hinge-loss error ($E$) is defined as,

$$E = \max_{j=1, \cdots, n} |e_j|$$  \hspace{1cm} (13)

Using the predicted output ($\hat{y}^t$), the predicted class label ($\hat{c}^t$) is obtained as shown below.

$$\hat{c}^t = \arg\max_{j=1, \cdots, n} (\hat{y}^t_j)$$  \hspace{1cm} (14)
4.2.2. Meta-Cognitive Sequential Learning Algorithm

Aim of the learning algorithm for a classifier is to determine the number of rules and its parameters to approximate the decision function as closely as possible. The meta-cognitive component of McFIS provides a mechanism to determine the number of rules and its corresponding parameters by controlling what-to-learn, when-to-learn and how-to-learn in a neuro-fuzzy inference system. McFIS controls the learning by monitoring the prediction error Eqn. (13) and spherical potential of the network to a given sample.

When a sample is presented to the network, McFIS controls the learning by activating one of the following strategies which help mimic the best reported human learning mechanism (refer Fig. 4).

![Fig. 4. Meta-cognitive learning mechanism in McFIS](image)

**Action a) Sample Deletion Strategy:** If the predicted class label is the same as the actual class label and the prediction error is below a threshold (\(E_d\)), then the current sample is similar to the knowledge already present in McFIS classifier and hence, deleted without being used in the learning process thereby, avoiding over-training and reducing the computational effort. The condition is given by

\[
\text{IF } c^t = \hat{c}^t \text{ AND } E \leq E_d \text{ THEN delete the sample.} \quad (15)
\]

In this study, \(E_d\) is chosen in the range \([0.05, 0.1]\). A higher value of \(E_d\) results in frequent deletion of training samples without being learnt, whereas a lower value results in fewer samples being deleted, resulting in over-training. This strategy addresses the what-to-learn aspect of meta-cognition.

**Action b) Sample Learning Strategy:** This strategy addresses how-to-learn aspect of meta-cognition. In McFIS, a sample is learnt by either adding/pruning a rule to the network or updating the parameters of the nearest rule using a decoupled extended Kalman filter.

In McFIS, a new rule is added to the network if the prediction error is higher than self-regulatory adding threshold and spherical potential calculated is lower than the novelty threshold. Spherical potential is a measure used often in kernel methods to determine the novelty of the data. Here, the novelty is determined based on the projection of an input feature (\(x^t\)) on to a hyper-dimensional feature space \(S\), i.e., \(x^t \rightarrow \phi\). Since, we are using Gaussian functions for projection, the hyper-dimensional feature space will be spherical in nature. In McFIS, the center (\(\mu\)) and width (\(\sigma\)) of the Gaussian rules describe the hyper-dimensional feature space \(S\). Let the center of the \(K\)-dimensional space be \(\phi_0 = \frac{1}{K} \sum_{i=1}^{K} \phi(\mu_i)\).

The spherical potential (\(\psi\)) of any sample (\(x^t\)) in the feature space is expressed as a squared distance from the hyper-dimensional mapping \(S\) centered at origin \(\phi_0\) and is given by

\[
\psi = -\frac{2}{K} \sum_{j=1}^{K} \phi(x^t, \mu_j^c) \quad (16)
\]

Since we are addressing the classification problem, the class-specific distribution plays a vital role and it influences the performance the classifier significantly. Hence, spherical potential is redefined as class-specific spherical potential and is given by

\[
\psi_c = -\frac{2}{K} \sum_{j=1}^{K^c} \phi(x^t, \mu_j^c) \quad (17)
\]

where, \(K^c\) is the number of rules associated with the class \(c\).

Condition for growing a new rule is given by

\[
E > E_a \quad \text{AND} \quad c^t \neq \hat{c}^t \quad \text{AND} \quad |\psi_c| < E_S \quad (18)
\]

where \(E_S\) is a class-specific novelty threshold and \(E_a\) is a self-adaptive meta-cognitive addition parameters. \(E_a\) is adapted by the following equation as given below

\[
E_a := (1 - \delta)E_a - \delta E \quad (19)
\]
\[ \delta \text{ being the slope control parameter set close to 0.} \]

If \( E_a \) is selected close to one, all misclassified samples will be used for rule addition. If one selects the threshold close to 2, then very few rules will be added and the resultant network may not approximate the decision surface. Hence, the suggested range for this threshold is [1.2, 1.7]. If one keeps the threshold \( E_S \) close 0, then the algorithm will not allow addition of rule. Similarly, if one keeps it close to 1, then all sample will be identified as novel sample. Hence, the class specific novelty threshold should be in the range [0.4, 0.7].

When a new rule is added the corresponding output weights (\( \alpha \)) are assigned such that the localization property of Gaussian rule could be exploited maximally. It is given by

\[ \alpha_{j,K+1} = \begin{cases} y_j^t = \frac{\sum_{i=1}^{K} \alpha_{i,j} \phi_i}{1 + \sum_{i=1}^{K} \phi_i}, & \text{otherwise} \\ 0, & y_j^t + \frac{\sum_{i=1}^{K} \alpha_{i,j} \phi_i}{} > 1 \end{cases} \]

where \( j = 1, 2, \ldots, n \).

The width of the new rule is assigned considering the inter-class and intra-class separation. Let \( nrS \) be the nearest rule in the intra-class (‘same class’) and \( nrI \) be the nearest rule in the inter-class (‘other classes’). They are defined as

\[ nrS = \arg \min_{l=c \forall j} \| x^t - \mu_{jr} \|; \quad nrI = \arg \min_{l \neq c \forall j} \| x^t - \mu_{jr} \| \]

Let the Euclidian distance between the nearest inter/intra rules be defined as

\[ d_S = \| x^t - \mu_{nrS} \|; \quad d_I = \| x^t - \mu_{nrI} \| \]

When the current sample \( (x^t) \) is closer to the nearest rule in the same class then the sample does not overlap with other classes. Then the width is assigned as

\[ s_{K+1} = \kappa \| x^t - \mu_{nrS} \| \]

When the current sample \( (x^t) \) is closer to the nearest rule in the inter-class then the sample overlaps with other classes. The width in this case is assigned as

\[ s_{K+1} = \eta \| x^t - \mu_{nrI} \| \]

where, \( \kappa \) determines the overlap between the new rule and nearest rule in the same class and \( \eta \) determines the overlap between the new rule and the nearest rule in the inter class. \( \kappa \) is usually chosen in the range [0.5, 0.9] and \( \eta \) in the range [0.1, 0.4].

When the maximum hinge error is greater than the self-adaptive threshold \( (E_l) \) and the predicted class label is the same as the actual class label then the parameters of the nearest rule in the same class are updated using a decoupled extended Kalman Filter algorithm. The update criterion is given by

\[ E \geq E_l \text{ AND } c^t = c^l \]

where, \( E_l \) is the self-adaptive update threshold. This threshold \( E_l \) is adapted based on the prediction error as:

\[ E_l := (1 - \delta) E_l - \delta E \]

(26)

If one selects the threshold \( E_l \) close to 1, then no sample will be used in learning and hence the resultant network do not approximate the decision function accurately. If \( E_l \) is selected close to 0, then all samples will be learnt, leading to over-fitting. Further details on parameter selection could be found in 61.

If the contribution of a rule \( (\beta_i) \) in the same class lesser than a pruning threshold \( E_p \) for \( N_w \) consequent samples in the same class then that rule is insignificant and can be removed from the network. The contribution of the rule \( (\beta_i) \) is measured as

\[ \beta_i = \frac{E}{\psi_c} \phi_i \max |\alpha_i|, \]

(27)

where \( E \) is the maximum hinge error, \( \psi_c \) is spherica potential of the actual class (c), \( \phi_i \) is the firing strength of the ith rule and \( \alpha_i \) is the output weight of the ith rule.

Action c) Sample Reserve Strategy: If the current sample does not satisfy either the growing condition or the update condition, then the sample is pushed to the rear-end of the training data stream. Since McFIS employs self-adaptive thresholds, these samples may be used at a later stage in the learning process.

In the next section, we provide a performance comparison of the developed method in accurately identifying various actions.

5. Performance Evaluation

In this section we provide a performance evaluation of the developed McFIS with optical flow features in recognizing human actions efficiently. For
performance analysis, two data sets are considered: Weizmann data set \(^\text{39}\) and KTH data set \(^\text{40}\) for human action recognition. For each of the data set, the performance is compared at frame level as well as sub-sequence level. For the performance evaluation of McFIS, the optical flow features extracted from the two data sets are compared with kernel Support Vector Machines (SVM) \(^\text{62}\). For SVM, the parameters \(C\) and \(\gamma\) are optimized using grid search technique. Both SVM and proposed McFIS are implemented in Matlab 7.11 on Windows environment with Pentium duo processor (2.6 GHz) and 4 GB memory. Performance of proposed McFIS classifier is compared with SVM on frame level and subsequence level features. In addition, the significance of optical flow features as feature vector for action recognition problem has been analyzed by considering the performance comparison with various other state-of-the-art methods reported in literature.

5.1. Recognition at Frame Level

The Weizmann\(^*\) data set contains 90 low resolution video sequences (180 × 144), where 10 actions are performed by 9 subjects. The background videos are also provided for all action sequences. The 10 actions include: bend, jumping-jack (jack), jump-forward-on-two-legs (jump), jump-in-place-on-two-legs (pjump), run, gallop-side-ways (side), skip, walk, wave-one-hand (wave1) and wave-two-hands (wave2). Totally 5036 frame level feature vectors are obtained from 10 action classes. For training, 3000 randomly selected frame level features are used and the remaining 2036 frames are used for testing. The performance of the classifier is evaluated using average and overall accuracies. For each action in KTH\(^\dagger\) data set, the sequences without scale change are considered. Totally, 10,398 frame level samples are obtained from 6 action classes, viz., walking, jogging, boxing, clapping, waving and running. Randomly chosen 5000 samples are used for training and tested using the remaining samples.

<table>
<thead>
<tr>
<th>Action</th>
<th>McFIS</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
</tr>
<tr>
<td>Bend</td>
<td>98.8</td>
<td>97.4</td>
</tr>
<tr>
<td>jack</td>
<td>98.8</td>
<td>96.1</td>
</tr>
<tr>
<td>Jump</td>
<td>99.6</td>
<td>97.8</td>
</tr>
<tr>
<td>pJump</td>
<td>99.2</td>
<td>92.3</td>
</tr>
<tr>
<td>Run</td>
<td>98.3</td>
<td>96.2</td>
</tr>
<tr>
<td>Side</td>
<td>100.0</td>
<td>95.5</td>
</tr>
<tr>
<td>Skip</td>
<td>94.8</td>
<td>85.5</td>
</tr>
<tr>
<td>Walk</td>
<td>100.0</td>
<td>99.3</td>
</tr>
<tr>
<td>Wave1</td>
<td>98.9</td>
<td>94.2</td>
</tr>
<tr>
<td>Wave2</td>
<td>98.7</td>
<td>93.4</td>
</tr>
</tbody>
</table>

Table 1. Performance comparison of frame level Weizmann data set

<table>
<thead>
<tr>
<th>Action</th>
<th>McFIS</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
</tr>
<tr>
<td>Boxing</td>
<td>78.0</td>
<td>79.6</td>
</tr>
<tr>
<td>Hand clapping</td>
<td>92.67</td>
<td>91.8</td>
</tr>
<tr>
<td>Hand waving</td>
<td>81.6</td>
<td>79.3</td>
</tr>
<tr>
<td>Jogging</td>
<td>75.4</td>
<td>62.9</td>
</tr>
<tr>
<td>Running</td>
<td>67.6</td>
<td>42.2</td>
</tr>
<tr>
<td>Walking</td>
<td>92.8</td>
<td>89.0</td>
</tr>
</tbody>
</table>

Table 2. Performance comparison of frame level KTH data set

The performance for individual action for Weizmann data set and KTH data set is given in table 1 and 2, respectively. In order to compare the results with regular classifiers, the same data set is used for training multi-class SVM, using lib linear package\(^\ddagger\). The performance comparison on SVM for Weizmann and KTH data set is given in table 1 and 2, respectively. From the tables it could be seen that, for Weizmann data set, SVM has an average training and testing efficiency of 65.8% and 69.9%. McFIS outperforms SVM classifier by about 30%. It should be noted that McFIS employs 450 rules and 90% of the total training samples to achieve this accuracy. Similarly for KTH data set, McFIS outperforms SVM by about 20%. In both these scenarios, the meta-cognitive components of McFIS along with the spherical potential has helped learn the information contained in the sample better.

5.2. Sub-sequence Level Recognition

For sub-sequence level recognition, in Weizmann data set there are 4385 sub-sequences from 10 action classes. For training, 3000 randomly selected sub-sequences are used and the remaining is used for test-
Human Action Recognition using McFIS

ing. Similarly for KTH data set there are 10,084 sub-sequences from 6 action classes. For training, 5000 randomly chosen sub-subsequence features are used and the performance of McFIS classifier is tested using the remaining sub-sequences.

The testing confusion matrix for Weizmann and KTH data set on sub-sequence level recognition is provided in table 3 and 4, respectively. In Weizmann data set the misclassification is between jumping-jack and jump-in-place-on-two-legs, which is visually similar. Similarly, skipping, jumping and galloping belong to the same group of actions and Waving of hands belong to a single group.

In case of KTH data set significant misclassification occur among jogging, running and walking classes, which are visually similar except for the speed factor.

Table 3. Confusion matrix: Weizmann Dataset on Sub-Sequence Level Features

<table>
<thead>
<tr>
<th>Action</th>
<th>Bend</th>
<th>Jack</th>
<th>Jump</th>
<th>pjJump</th>
<th>Run</th>
<th>Side</th>
<th>Skip</th>
<th>Walk</th>
<th>Wave1</th>
<th>Wave2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bend</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jack</td>
<td>0</td>
<td>99</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jump</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>pjJump</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>99</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Run</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Side</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Skip</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>97</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Walk</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Wave1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>99</td>
</tr>
<tr>
<td>Wave2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>99</td>
</tr>
</tbody>
</table>

Table 4. Confusion matrix: KTH Dataset on Sub-Sequence Level Features

<table>
<thead>
<tr>
<th>Action</th>
<th>Boxing</th>
<th>Clapping</th>
<th>Waving</th>
<th>Jogging</th>
<th>Running</th>
<th>Walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxing</td>
<td>97</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clapping</td>
<td>1</td>
<td>98</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Waving</td>
<td>1</td>
<td>3</td>
<td>96</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jogging</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>89</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Running</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>78</td>
<td>2</td>
</tr>
<tr>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>78</td>
<td>97</td>
</tr>
</tbody>
</table>

Table 5. Performance comparison of sub-sequence level Weizmann data set

<table>
<thead>
<tr>
<th>Action</th>
<th>McFIS</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
</tr>
<tr>
<td>Bend</td>
<td>99.8</td>
<td>100</td>
</tr>
<tr>
<td>Jack</td>
<td>99.4</td>
<td>99.3</td>
</tr>
<tr>
<td>Jump</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>pJump</td>
<td>100</td>
<td>99.4</td>
</tr>
<tr>
<td>Run</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Side</td>
<td>99.6</td>
<td>100</td>
</tr>
<tr>
<td>Skip</td>
<td>99.2</td>
<td>96.9</td>
</tr>
<tr>
<td>Walk</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Wave1</td>
<td>99.7</td>
<td>100</td>
</tr>
<tr>
<td>Wave2</td>
<td>99.7</td>
<td>99</td>
</tr>
</tbody>
</table>

Table 6. Performance comparison of sub-sequence level KTH data set

<table>
<thead>
<tr>
<th>Action</th>
<th>McFIS</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
</tr>
<tr>
<td>Boxing</td>
<td>96.6</td>
<td>69.8</td>
</tr>
<tr>
<td>Hand clapping</td>
<td>97.6</td>
<td>67.5</td>
</tr>
<tr>
<td>Hand waving</td>
<td>96.2</td>
<td>86.8</td>
</tr>
<tr>
<td>Jogging</td>
<td>89.4</td>
<td>60.7</td>
</tr>
<tr>
<td>Running</td>
<td>77.1</td>
<td>31.9</td>
</tr>
<tr>
<td>Walking</td>
<td>96.7</td>
<td>66.2</td>
</tr>
</tbody>
</table>

Table 7. Performance comparison of sub-sequence level KTH data set

<table>
<thead>
<tr>
<th>Action</th>
<th>McFIS</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
</tr>
</tbody>
</table>

The performance for individual action at sub-sequence level for Weizmann and KTH data set on state-of-the-art SVM classifier is given in tables 5 and 6, respectively. For Weizmann data set, it could be seen that the testing performance of McFIS is significantly better than that of SVM. It should be noted that McFIS attains this performance employing about 90% of the training samples and 250 rules alone. Similarly, for KTH data set it could be seen that, McFIS outperforms SVM. The poor performance of SVM is due to the fact that it solves multi-category classification problems as several binary classification problems, whereas the action recognition problem considered here is a multi-class classification problem.

5.3. Comparison of Sub-sequence Level Recognition Performance

Next, we compare the action recognition results of the proposed approach with some of the state-of-the-art methods for Weizmann and KTH data sets. In this paper simple optical flow based features are employed whereas 63 and 59 employ silhouette based global level information. These information are prone to noise and are difficult to extract. In 63 bag of words model is employed where, the motion descriptors are obtained after tracking and stabilizing the persons in the video. Authors in 64 employed quantized spatio-temporal features which employs figure-centric information about the local spatio-temporal distribution to compute statistics of pairwise co-occurring visual words within its neighbourhoods.

The comparison is for Weizmann data set is given in table 7 and for KTH data set in table 8. It could be seen that, the performance of the proposed method is better than the state-of-the-art approaches reported in literature.
Table 7: Comparison of proposed method with state-of-the-art methods for Weizmann data set

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Optical flow</th>
<th>Silhouette</th>
<th>Space-time shapes</th>
<th>Kinematic features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>McFIS</td>
<td>SVM</td>
<td>Ref. 53</td>
<td>Ref. 39</td>
</tr>
<tr>
<td>Avg. Testing Accuracy</td>
<td>99.46%</td>
<td>87.34%</td>
<td>96.74%</td>
<td>97.83%</td>
</tr>
</tbody>
</table>

Table 8: Comparison of proposed method with state-of-the-art methods for KTH data set

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Optical flow</th>
<th>Silhouette</th>
<th>Space-time shapes</th>
<th>Kinematic features</th>
<th>Quantized local spatio-temporal features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>McFIS</td>
<td>SVM</td>
<td>Ref. 53</td>
<td>Ref. 40</td>
<td>Ref. 65</td>
</tr>
<tr>
<td>Testing Accuracy</td>
<td>95.4%</td>
<td>63.8%</td>
<td>91.2%</td>
<td>71.7%</td>
<td>87.7%</td>
</tr>
</tbody>
</table>

6. Conclusion

In this paper a meta-cognitive neuro-fuzzy inference system was employed for solving human action recognition problem. Since, optical flow vectors can extract simple dynamic features hierarchically from local pixel level to global object level, optical flow between two image planes, represented as two real valued features was employed. The relationship between optical flow features and action label are approximated using McFIS classifier. The meta-cognitive learning algorithm decides on what-to-learn, when-to-learn and how-to-learn in a neuro-fuzzy inference system and accordingly removes it without learning, learns it by either growing/pruning of rule or updating the parameter or reserve for future use. The sequential learning algorithm of McFIS is controlled and monitored by the meta-cognitive components which uses class-specific, knowledge based criteria along with self-regulatory thresholds to decide on one of the following strategies: a) sample deletion b) sample learning and c) sample reserve. The performance of proposed method is compared with standard SVM and other state-of-the-art methods in literature. The results indicate superior performance of McFIS in recognizing human actions compared the results reported in the literature.

7. References


