Received August 31, 2017, accepted October 2, 2017, date of publication November 2, 2017,
date of current version December 22, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2769140

Soccer Video Structure Analysis by Parallel Feature Fusion Network and Hidden-to-Observable Transferring Markov Model

MEHRNAZ FANI[1], (Member, IEEE), MEHRAN YAZDI[1], (Member, IEEE), DAVID A. CLAUSI[2], (Senior Member, IEEE), AND ALEXANDER WONG[2], (Senior Member, IEEE)
[1]Department of Electrical and Computer Engineering, Shiraz University, Shiraz 51154-71348, Iran
[2]Department of System Design, University of Waterloo, Waterloo, ON N2L 3G1, Canada

Corresponding author: Mehrnaz Fani (fani.mehrnaz@shirazu.ac.ir)

ABSTRACT Automated analysis of broadcast soccer game video is a challenging computer vision problem. Prior to performing high-level analysis (such as event detection), accurate classification of shot views and play–break segmentation are required to analyze the structure of soccer video. A novel deep network called parallel feature fusion network (PFF-Net) combines local and full-scene features to produce accurate shot view classification based on camera zoom and out-of-field status. Then, a novel hidden-to-observable Markov model (H2O-MM) is introduced to determine play/break status of the shots. Testing is performed using a variety of professional broadcast soccer videos. Variations of the PFF-Net are considered and compared with four existing methods where the PFF-Net demonstrates superior performance (92.6%). The H2O-MM has the accuracy of 98.7% for play–break segmentation, which is an improvement over two existing hidden Markov models. The new methods provide improved temporal labeling of broadcast soccer videos, which can be used to further improve overall automated event detection.

INDEX TERMS Parallel feature fusion network, hidden to observable Markov model, shot view classification, play-break segmentation, soccer video analysis, stacked autoencoders.

I. INTRODUCTION

Automatic analysis of sport videos for evaluating performance of players or adding some meta-data to broadcast sport videos is a current, challenging research topic [1]–[5]. Soccer, as a popular worldwide sport, specifically appeals to analyzers and researchers [1], [6]–[8]. Published research on soccer video analysis has mainly focused on automatic event detection, retrieval, and summarization of broadcast videos [9]–[14].

However, as a prerequisite for these high-level tasks, a series of lower level analyses is required to break the full-length structure of broadcast soccer videos into useful temporal units such as shot and play-break segments (or event segments), and extract a hierarchy of features from these units. Performing such low/mid-level analyses by designing and implementing new networks is the main focus of this article.

A “shot” in video can be described as a continuous sequence of frames taken by a non-stop camera operation [15]. A shot in a soccer video is in “play” state when the game is proceeding and a “break” occurs at any game stoppage e.g., goal, foul, corner, etc. [9]. So consecutive play and break shots mark an event segment in a soccer video and is referred to as a play-break segment. Shot views are useful features for play-break segmentation of soccer videos, and can be divided into four main types: close-up, medium, long, (as three in-field views) and out-of-field view. An example of each shot view is provided in Fig. 1, and a brief description of each is given here:

- **Close-up shot view:** a view that shows an upper-body of a player and typically indicates a break in the game based on some preceding event [16].

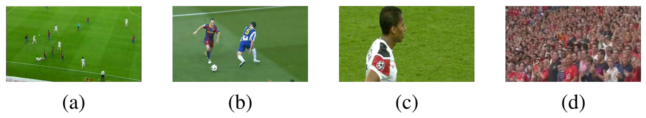


FIGURE 1. Instances of typical shot views in soccer videos: (a) long, (b) medium, (c) close-up, and (d) out-of-field views. Ratio of green region from (a) to (d) decreases, this ratio has been used as a key feature for shot view classification in most of the previous published research.

- **Medium shot view:** a view that ranges from showing a group of players to a whole player body used in play or break state for a closer view of the players [12].
- **Long shot view:** a full or near-full view of the field where the player size is much smaller than the image size [16] which is used to show the nature of team plays.
- **Out-of-field shot view:** a view that predominantly displays audiences or the team sidelines to capture people reactions to the game [12].

Shot boundary detection, shot view classification, and play-break segmentation are three consecutive tasks that make subsequent analyses of soccer videos easier. A considerable research effort in video analysis is devoted to shot boundary detection [15], [17]–[19]. A part of research in context of soccer video analysis focuses on shot view classification, and play-break segmentation [9], [11], [12], which are traditionally performed through extraction of handcrafted features from video shots. However, high capabilities of deep networks in solving different machine learning problems encourages their use in soccer video analysis as well.

The two main focus points of this article are on shot view classification and play-break segmentation in broadcast soccer videos by employing new networks. As such, the main contributions of this article are:

- Designing and implementing a novel deep network, named as parallel feature fusion network (PFF-Net), with innovative feature extraction and integration for shot view classification in soccer video.
- Introducing a hidden Markov model, named as hidden-to-observable transferring Markov model (H2O-MM), with an innovative learning and inference mechanism, for play-break segmentation of soccer videos.

II. BACKGROUND

The background section is composed of three subsections: shot view classification (Subsection A), play-break classification (Subsection B), and existing deep networks used for soccer video analysis (Subsection C).

A. SHOT VIEW CLASSIFICATION IN SOCCER

In most of shot view classification methods in soccer genre, dominant color ratio has been considered as a key feature. It is assumed that the grass color (a tone of green) is the dominant color in long shots, while the grass ratio decreases in other shot views and reduces to zero in out-of-filed and some of the close-up shots, (see Fig. 1).

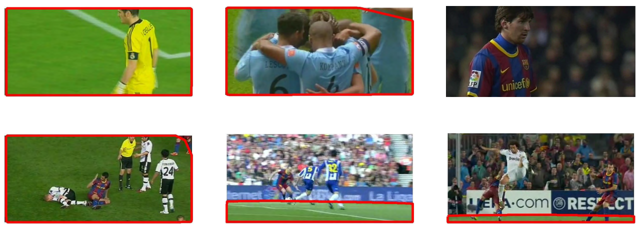


FIGURE 2. Top tow, instances of close-up views, and bottom row, instances of medium views, with different proportions of green regions, show that green region ratio alone is not a reliable feature for shot view classification and to increase the accuracy of classification other features should be extracted from video shots.

Ekin *et al.* [12] assume that if the grass ratio is smaller than a threshold, shot view is close-up or out-of-field; otherwise, it would be either long or medium. To decide between the two latter views first, they partition the grass region according to Golden Section spatial composition rule [20]. Then, they classify the shot views by fitting two Gaussian distributions for long and medium views on extracted features from grass partitions and using a Bayesian classifier.

Sigari *et al.* [21] perform shot view classification by detecting the ratio of grass and skin in the HSV color space, and employing a threshold-based heuristic. As a result, they classify shots to close-up, out-of-field and in-field views.

The grass ratio is not always insignificant in close-up view and has a varying proportion in medium view (see Fig. 2). Hence, using the grass ratio as the only feature for shot view classification will not always be effective.

To increase the accuracy of shot view classification more features are employed in other works. Tavassolipour *et al.* [9] besides the grass ratio use size of the largest object in the field and classify the shot views by means of a decision tree.

Sigari *et al.* [22] use the same features as used by Tavassolipour, and classify shot views into three classes, namely long, medium and other views, by cascading two support vector machines (SVMs) [23].

Tong *et al.* [16] use four features: grass-ratio, texture (gray level co-occurrence matrix), head region (a skin-toned entity) and object scale (height of the object to the height of the field). By analyzing the statistics from the training images, the authors determined some threshold values for the extracted features, and constructed a decision tree to find shot views.

Sadlier and O'connor [6] hypothesize that a frame is in close-up view if the face of a player (using skin tone) is located in the top-middle region of the frame and a player jersey (assumed to be monochromatic) covers most of the area of the bottom middle region. They also postulate that frames in out-of-field shots contain many details (i.e., include many high frequency blocks). They find high frequency image blocks using DCT coefficients, and determine out-of-field frames.

The published research on shot view classification in soccer mainly relies on extracting handcrafted features from video. The extracted features need to be invariant to some irrelevant variations such as lighting and pose of players, but sensitive to size of a player, which is difficult to extract due to exceptions such as overlapping of the players' bodies. Hence, some feature extractors are required that are reliable and can be learned automatically, instead of being fully engineered, [24]. This purpose is pursued in this article.

B. PLAY-BREAK SEGMENTATION IN SOCCER

After shot views are determined, the next step is to determine play/break state of each shot.

Tjondronegoro and Chen [1] apply some heuristic rules on shot views to classify each shot to play or break. Tjondronegoro *et al.* [25] show that using features such as dominant color ratio and motion intensity, when combined with rule-based detection techniques, could be effective in play-break segmentation.

Other research perform play-break segmentation using hidden Markov models [9], [11]. Tavassolipour *et al.* [9] use an hidden Markov models (HMMs) with two hidden states for play and break, and four observation states for shot views, as the input features. After learning the model parameters (i.e., state transitions and observation probabilities), in the test phase, a label sequence (a sequence of play/breaks) is generated using Viterbi decoding.

Xie *et al.* [11] use an HMM structure quite similar to the model employed by Tavassolipour *et al.* [9] except for using dominant color ratio and motion intensity as the observation states and being frame-based (not shot-based). So, play or break state is determined for all frames and not just the key-frame of a shot, which leads to significant computations.

C. DEEP STRUCTURES USED IN SOCCER ANALYSIS

Based on Subsections II.A and II.B, existing shot view classification and play-break segmentation methods focus on using handcrafted features such as dominant color ratio and skin detection. However, recently, deep architectures with their automatic feature extraction abilities have substituted the classic handcrafted feature extraction methods and have improved performance in many image processing applications [26]–[28]. But using deep networks for soccer video analysis is still in its infancy. Here, research applying deep networks on soccer (or related team sports) videos are reviewed.

Wagenaar *et al.* [29] use deep convolutional neural networks to detect goal-scoring opportunities in soccer. They employ data of tracked players and balls in 29 soccer matches, and achieved accuracy of 67.1% using GoogleNet. Park and Cho [30] use a CNN structure to perform shot view classification for basketball. They classify shot views into close-up and long views and gain accuracy of 99.23%. Gerke *et al.* [31] uses a CNN-based deep network to automatically recognize jersey numbers from soccer videos and obtain the accuracy of 83%.

Deep networks have shown strong success in extracting features which are difficult to be handcrafted [24], and have been used for addressing many vision problems. The

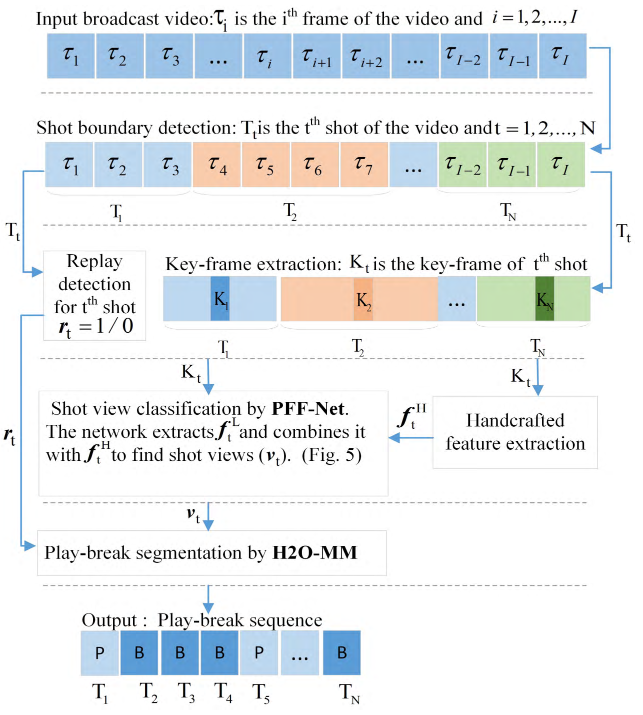

Input broadcast video: τ_i is the i^th frame of the video and $i = 1,2,...,I$
τ_1 τ_2 τ_3 ... τ_i τ_{i+1} τ_{i+2} ... τ_{I-2} τ_{I-1} τ_I
Shot boundary detection: T_t is the t^th shot of the video and $t = 1,2,...,N$
τ_4 τ_5 τ_6 τ_7
T_1 T_2 T_N
T_t
Replay detection for t^th shot $r_t = 1/0$
Key-frame extraction: K_t is the key-frame of t^th shot
K_1 K_2 K_N
K_t
Shot view classification by PFF-Net. The network extracts f_t^L and combines it with f_t^H to find shot views (v_t). (Fig. 5)
f_t^H
Handcrafted feature extraction
r_t
v_t
Play-break segmentation by H2O-MM
Output : Play-break sequence
P B B B P ... B
T_1 T_2 T_3 T_4 T_5 T_N

FIGURE 3. Designed framework for soccer video structure analysis. Broadcast soccer video is divided to shots [17], replay detection is performed for each shot [9], and key-frame of each shot is determined [32]. For each key-frame, handcrafted features (f_t^H) are extracted and used as input with raw key-frames (K_t) to the PFF-Net for shot view classification. Shot view (v_t) and replay (r_t) labels are used as input to the H2O-MM for play-break segmentation.

only known attempt for classifying shot views using deep networks is performed by Park and Cho [30] to discriminate between close-up and long views, in basketball videos, by a CNN.

Therefore, in this article, in the first stage, a novel deep network (PFF-Net), based on the nature of the scene in soccer video, is designed to classify four shot views with a high accuracy. The PFF-Net is relatively a small-sized deep network (including six layers) whose principle building modules work independently and are put in parallel fashion. The PFF-Net combines the automatically extracted local features with the handcrafted features, that describe a whole frame.

In the second stage, a new Markov model is introduced (H2O-MM) that employs the shot views for subsequent play-break segmentation of soccer videos. This model uses a novel iterative mechanism to increase the accuracy of inference by providing a feedback from hidden states to the observable variables. A randomness is also enforced to the model to avoid the early convergence to a local solution.

Here, organization of this article is provided. In Section III, an overview of proposed system for analyzing structure of soccer video is provided. In Section IV, handcrafted features are extracted from video key-frames. In Section V, architecture of PFF-Net for shot view classification is explained.

In Section VI, the H2O-MM model for play-break segmentation is described. In Section VII, experimental results are reported.

III. SYSTEM OVERVIEW FOR SOCCER VIDEO STRUCTURE ANALYSIS

A framework for broadcast soccer video structure analysis with the purpose of shot view determination and play-break segmentation is introduced and shown in Fig. 3.

First, shot boundaries are detected using an existing method [17] that employs histogram thresholding and SVM classifier for this purpose. So, the input video is divided into N shots. Then, it is determined if each shot is a replay shot or not (i.e., $\boldsymbol{r}_t = 1/0$). Replays in broadcast sport videos are usually placed between two graphical logos, and can be detected by detecting the logos. Here the procedure by Tavassolopour *et al.* [9] for logo detection and replay determination is employed. Next, key-frame of t^{th} shot, K_t for $t = 1, 2, .., N$, (a representative frame that reflects content of that shot) is extracted using a threshold on histogram differences of consecutive frames [32]. Next, a vector of handcrafted features, $\boldsymbol{f}_t^H$, is extracted from each key-frame K_t based on the method explained in Section IV.

Key-frames and feature vectors are fed into the PFF-Net, whose design and implementation are given in Section V, and shot views are determined. Finally, shot view ($\boldsymbol{v}_t$) and the replay label of the t^{th} shot ($\boldsymbol{r}_t$) are used as input to the H2O-MM for play-break segmentation (detailed in Section VI).

IV. HANDCRAFTED FEATURE EXTRACTION

Using the t^{th} key-frame, $\boldsymbol{f}_t^H$ is formed by extracting three handcrafted features, which are adopted from a successful shot view detection method [9], based on the following steps.

- Grass regions are detected using global thresholding [33] in RGB space and a convex hall is used to surround each grass region [9] (see Fig. 4).
- The area of the largest non-grass object within the convex hall is determined by use of connected components [33].
- Three handcrafted features are computed: **1.** The ratio of the green area (soccer field) to the whole image area, ($grn2im$), **2.** The ratio of the biggest object area in the green region to the image area ($obj2im$), and **3.** The ratio of the biggest object area in the green region to the green area ($obj2grn$). So, $\boldsymbol{f}_t^H = [grn2im, obj2im, obj2grn]^T$.

V. PFF-NET FOR SHOT VIEW CLASSIFICATION

The general architecture of PFF-Net is shown in Fig. 5. The PFF-Net has a parallel structure of stacked autoencoders (SAEs) for extracting local features and consolidating them with the handcrafted features for shot view classification. The j^{th} replica of SAE, along with a softmax layer on its top, is referred to as the j^{th} parallel network (PN^j).

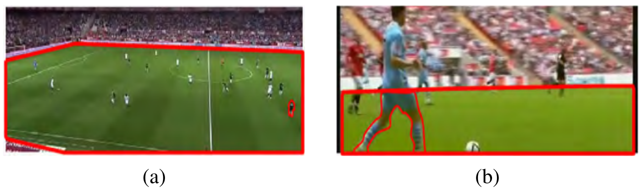


FIGURE 4. Examples of two key-frames in which convex hull around the soccer filed and the region enclosing the biggest object in the field are outlined. The outlined regions are employed for handcrafted feature extraction.

In designing the architecture of deep networks such as CNNs it is assumed that a feature detector which is learned over some small patches of image can be effectively applied anywhere in the image [34]. However, for designing the PFF-Net we hypothesize that feature detectors are more effective if specialized to describe a group of similar image patches rather than the entire image. Such local feature detectors could extract more content-representative features from patches. Also, as Hinton and Salakhutdinov have shown [35], stacked autoencoders are able to learn lower-dimensional representation of their input data which is better than the representation that PCA can produce. Hence, PNs embedded in PFF-Net use SAEs to learn distinct models for separate groups of input patches and provide lower-dimensional representation of them.

The PFF-Net is composed of three main components. **Component 1** is local data transformation layer that inputs key-frames, divides each into four horizontal strips and transforms data of each strip into its reciprocal eigenspace (Subsection A). **Component 2** includes four parallel pipelines of SAEs that extract local features, $\boldsymbol{f}^L$, from transformed data and combine them with the related handcrafted feature vector, $\boldsymbol{f}^H$, (Subsection B). **Component 3** is the LG-classification that performs shot view classification in local and global stages via local softmaxes and a Bayesian layer (Subsection C).

Here a concise review on stacked autoencoders, and softmax classifier as building blocks of the PFF-Net is provided to help understanding the network architecture.

An autoencoder [34] is a neural network with an unsupervised learning algorithm that encodes the input vector $\boldsymbol{x}_i \in \mathbb{R}^m$ into a hidden representation $\boldsymbol{h}_i \in \mathbb{R}^k$ and decodes the latent representation into the output reconstruction representation $\hat{\boldsymbol{x}}_i \in \mathbb{R}^m$. The network uses a nonlinear mapping, such as a sigmoid function, for transformations and employs a backpropagation algorithm to derive the output values to approximate the input values by adjusting the network weights. When the number of hidden units is smaller than the number of input/output units, the network is forced to learn a compressed representation of the input. If the sparsity constraint is also imposed on the hidden units, latent representation would be even more compressed, and linear separability in the latent space would be encouraged. The objective function of an existing autoencoder with sparsity constraint and L_2 regularization term can be written

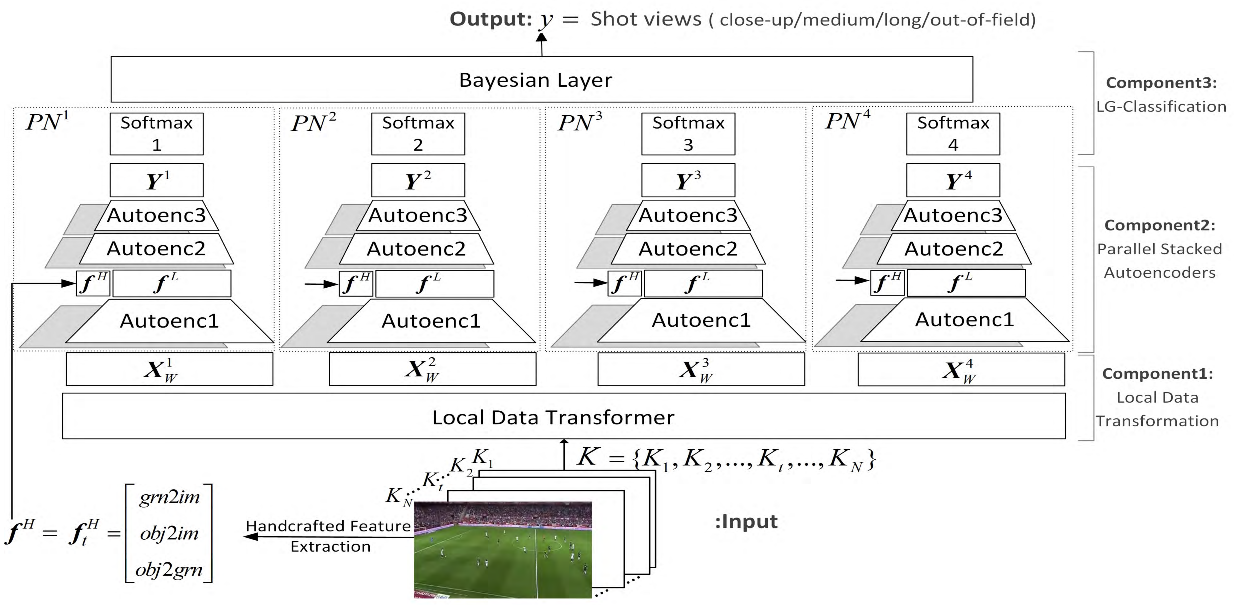

Output: y = Shot views (close-up/medium/long/out-of-field)
Bayesian Layer
Component3: LG-Classification
PN^1
PN^2
PN^3
PN^4
Softmax 1
Softmax 2
Softmax 3
Softmax 4
Y^1
Y^2
Y^3
Y^4
Autoenc3
Autoenc2
f^H
f^L
Autoenc1
Component2: Parallel Stacked Autoencoders
X_W^1
X_W^2
X_W^3
X_W^4
Component1: Local Data Transformation
Local Data Transformer
$K = \{K_1, K_2, ..., K_t, ..., K_N\}$
K_1
K_2
K_t
K_N
$f^H = f_t^H = \begin{bmatrix} grn2im \\ obj2im \\ obj2grn \end{bmatrix}$
Handcrafted Feature Extraction
:Input

FIGURE 5. Architecture of the PFF-Net is composed of three main components: **Component1**: The local data transformer performs strip-wised ZCA whitening on blocks of key-frames. **Component2**: Parallel stacked autoencoders extract local features from transformed blocks of strips, combine them with handcrafted features and provide a lower dimensional representation of the blocks. **Component3**: LG-classifier determines shot view of a key-frame, by first making decision for each block using local softmaxes and then global decision for the whole key-frame via Bayesian layer.

as [34]

$$J = \sum_{i=1}^{n} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|_2^2 + \beta \sum_{q=1}^{k} KL(\rho \| \hat{\rho}_q) + \lambda \sum_{l=1}^{L} \sum_{q=1}^{k} \sum_{p=1}^{m} (w_{qp}^{(l)})^2 \quad (1)$$

$$KL(\rho \| \hat{\rho}_q) = \rho log \frac{\rho}{\hat{\rho}_q} + (1-\rho) log \frac{(1-\rho)}{(1-\hat{\rho}_q)}. \quad (2)$$

In Eq. 1, the first term is the mean-squared reconstruction error over n training samples. The second term is the sparsity regularization term with coefficient β. Here, ρ is the sparsity proportion which determines the desired proportion of training samples to which a neuron reacts, and $\hat{\rho}_q$ is the average activation of hidden unit q over the entire training set. To enforce the sparsity, Kullback-Leibler divergence (KL) between ρ and $\hat{\rho}_q$ is employed as the penalty function, based on Eq. 2. The last term is the L_2 regularization term with coefficient λ, which prevents the undesirable increase of the network weights ($w_{qp}^{(l)}$) [36], where l denotes the layer index. The autoencoder is trained when the network parameters are optimized by minimizing the objective function J over the training set.

A stacked autoencoder (SAE) [34] can be constructed by stacking further unsupervised feature learning layers, in a way that the hidden representation of the first autoencoder is used as the input of the second autoencoder and so on. An SAE is able to learn a lower-dimensional representation of input data which is better than the representation that PCA can give [35].

A softmax classifier [34] can be utilized in the last layer of an SAE structure. It is used for supervised fine-tuning the network and finally classifying the inputs. By fitting a model on the training set, softmax estimates the probability of a sample being assigned to a particular class c, where c is one of C labels. So, for a test pair $(\boldsymbol{x}, y)$, softmax gives a C-dimensional vector of probabilities based on Eq. 3, where $\Theta = \{\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \ldots, \boldsymbol{\theta}^C \in \mathbb{R}^m\}$ is the set of model parameters:

$$\begin{bmatrix} P(y=1|\boldsymbol{x},\Theta) \\ P(y=2|\boldsymbol{x},\Theta) \\ \vdots \\ P(y=C|\boldsymbol{x},\Theta) \end{bmatrix} = \frac{1}{\sum_{c=1}^{C} \boldsymbol{\theta}^{c^T}} \begin{bmatrix} exp(\boldsymbol{\theta}^{1^T}\boldsymbol{x}) \\ exp(\boldsymbol{\theta}^{2^T}\boldsymbol{x}) \\ \vdots \\ exp(\boldsymbol{\theta}^{C^T}\boldsymbol{x}) \end{bmatrix} \quad (3)$$

Suppose that we have a training set of n labeled samples, $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$, where $\boldsymbol{x}_i \in \mathbb{R}^m$ is a training sample and y_i is its corresponding class label. Model parameters can be found by minimizing the cost function of Eq. 4 with an iterative optimization procedure [34]:

$$J(\Theta) = -\left[\sum_{i=1}^{n} \sum_{c=1}^{C} 1\{y_i = c\} log \frac{exp(\boldsymbol{\theta}^{c^T}\boldsymbol{x}_i)}{\sum_{c=1}^{C} exp(\boldsymbol{\theta}^{c^T}\boldsymbol{x}_i)} \right]. \quad (4)$$

Here, $1\{y_i = c\}$ is an indicator function [34] which has the value 1 if $y_i = c$ and 0 otherwise.

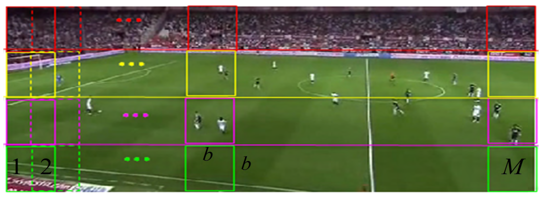


FIGURE 6. Horizontal partitioning of a key-frame and extracting *M* (*b* × *b*) blocks from each strip. This partitioning is used to form strip-wise similarity groups.

A. LOCAL DATA TRANSFORMATION

The first component of the PFF-Net is local data transformer that exploits the strip-wise similarities found in soccer video key-frames. Here, first, in Part 1, the existence of this similarity is shown mathematically, and then, in Part 2, this similarity is employed for designing the local data transformer.

1) STRIP-WISE SIMILARITY IN SOCCER VIDEO FRAMES

As illustrated in Fig. 6, each key-frame (with size 320 × 120) is divided into four horizontal strips with equal heights and $b \times b = 30 \times 30$ overlapping blocks (with step size of 20 pixels) are extracted from each strip. Here, it is assumed that blocks of the same horizontal strip are similar enough to be represented by a common model and a single feature-detector could be assigned to them.

The intuition behind this assumption is that, in a given shot of a soccer game, the dominant motion (combination of camera and objects motions) is stronger in the horizontal direction. For instance, Fig. 7 shows the average motion of blocks over 1000 frames in a soccer shot, where horizontal components of motion vectors are bigger than the vertical components. So, we expect the blocks in horizontal vicinities in successive frames to be similar. Also, in a single frame of a shot, such as the long shot shown in Fig. 6, a similarity in content of image blocks within the same strip exists, which is due to the arrangement of the soccer stadium and the camera vantage point. In this figure, in the top strip, audiences could be seen. In the second strip, soccer field and some very small-sized players are observable. In the third strip, which is closer to the camera, size of players has increased slightly. Finally, in the bottom strip, soccer field and the marginal lines are observable.

So, it is assumed that key-frames with the same shot view, in soccer, have similar patterns in their horizontal strips. To mathematically demonstrate this, a similarity measurement is computed that shows intra-strip similarity is more than inter-strip similarity. First, blocks extracted from all shot views are whitened [34], and their ring projection transform (RPT) [37] is computed. This transformation is illustrated in Fig. 8, and the obtained feature vector for block Bj is given in Eq. 5

$$RPT_{Bj} = \left[RPT(0), \ldots, RPT(r), \ldots, RPT(R)\right] \tag{5}$$

where, $RPT(r)$ is the mean of pixel values on circle r. Then, normalized cross correlation (NCC) is used to measure similarity between each pair of blocks (i.e., B_i, B_j), as given in

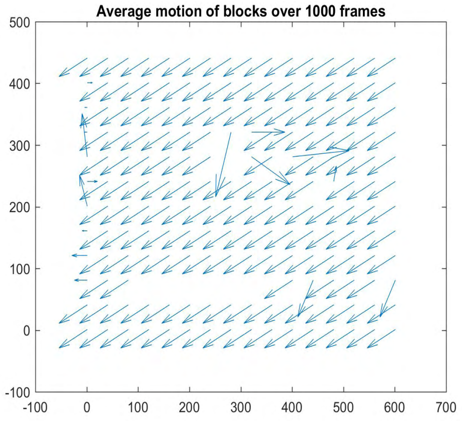
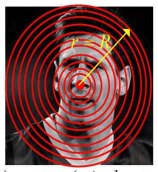


FIGURE 7. Average motion of blocks over 1000 frames of a soccer video shot, showing dominant motion in soccer is in horizontal direction. Sum of horizontal component of the motion vectors = −45223, sum of vertical components of motion vectors = −28048. Therefore, it could be interpreted that each block is more similar to its horizontal neighboring blocks of other frames compared to its vertical neighbors. As such, frame partitioning has been performed only in horizontal direction to form groups of similar blocks.



$RPT(r) = mean(pixels\ on\ circle\ r)$

FIGURE 8. Visual demonstration of ring projection transform (RPT) for different radiuses, where *RPT*(*r*) is the mean of all pixels on a circle with radius *r*.

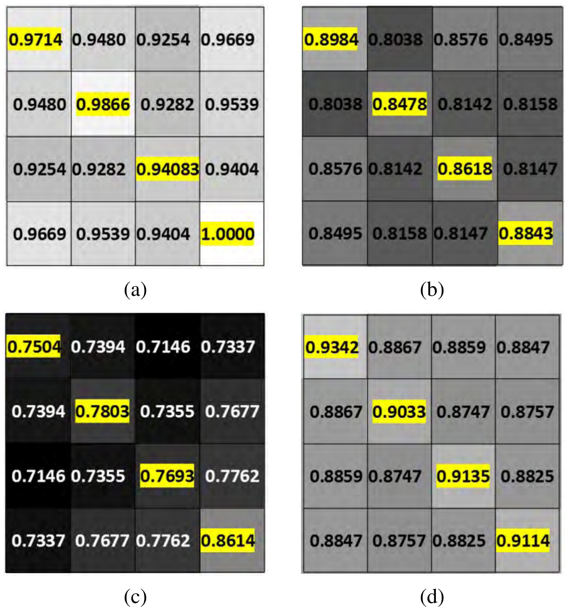
Eq. 6. Similarity between strips A and B is defined as the average of NCC for all pairs of blocks in the two strips (see Eq. 7). By computing the similarity between each possible pair of strips, a similarity matrix is achieved.

$$NCC(B_i, B_j) = \frac{\sum_{r=0}^{R}(RPT_{Bi} - \overline{RPT}_{Bi})(RPT_{Bj} - \overline{RPT}_{Bj})}{\sqrt{\left(\left[\sum_{r=0}^{R}(RPT_{Bi} - \overline{RPT}_{Bi})\right]^2 \left[\sum_{r=0}^{R}(RPT_{Bj} - \overline{RPT}_{Bj})\right]^2\right.}} \tag{6}$$

$$Sim(A, B) = \frac{1}{2n}\sum_{B_j \in B}\sum_{B_i \in A} NCC(B_i, B_j). \tag{7}$$

A similarity matrix per shot view is computed using $n = 1000$ random blocks of size $b \times b$ for each strip. The matrices are normalized by dividing their entries by maximum similarity value of all classes and demonstrated in Fig. 9. The normalized matrices are denoted as $\boldsymbol{M}_{sim1}$, $\boldsymbol{M}_{sim2}$, $\boldsymbol{M}_{sim3}$, and $\boldsymbol{M}_{sim4}$.

As shown in Fig. 9, the average similarity, (i.e., Sim), is highest in close-up and least in long view. Because in close-up, many blocks belong to the same object that is closer to



(a)

0.9714	0.9480	0.9254	0.9669
0.9480	0.9866	0.9282	0.9539
0.9254	0.9282	0.94083	0.9404
0.9669	0.9539	0.9404	1.0000

(b)

0.8984	0.8038	0.8576	0.8495
0.8038	0.8478	0.8142	0.8158
0.8576	0.8142	0.8618	0.8147
0.8495	0.8158	0.8147	0.8843

(c)

0.7504	0.7394	0.7146	0.7337
0.7394	0.7803	0.7355	0.7677
0.7146	0.7355	0.7693	0.7762
0.7337	0.7677	0.7762	0.8614

(d)

0.9342	0.8867	0.8859	0.8847
0.8867	0.9033	0.8747	0.8757
0.8859	0.8747	0.9135	0.8825
0.8847	0.8757	0.8825	0.9114

FIGURE 9. Normalized similarity matrices, (M_{simc}, c = 1, 2, 3, 4) for the four shot views, with their average similarity value, *Sim*. The main diagonals of the matrices have higher values. This indicates that intra-strip similarity for key-frames of all shot views is higher than inter-strip similarity, which justifies formation of strip-wise similarity groups. (a) M_{sim1}, *Sim*= 0.9515, (close-up). (b) M_{sim2}, *Sim*=0.8377, (medium). (c) M_{sim3}, *Sim*= 0.7560, (long). (d) M_{sim4}, *Sim*=0.8902, (out-of-field).

the camera and covers most of the frame. In medium, and long views distance between subjects and camera increases and a more open view of the scene is observable, so average similarity of blocks decreases. For out-of-field view, average similarity is high due to the repetitive pattern of audiences that appears all over a frame (see Fig. 1).

For all shot views, the main diagonals of the matrices have higher values. This indicates that intra-strip similarity is higher than inter-strip similarity which justifies the assumption of forming strip-wise similarity groups.

2) LOCAL DATA TRANSFORMER

The first component of the PFF-Net is the local data transformer, see Fig. 10, which is designed based on the local strip-wise similarity found in Part 1. Given the intra-strip similarity, the blocks can be effectively transformed to the same subspace.

Therefore, first, N training key-frames are split into four strips and M overlapping blocks (with size $m = b \times b$ pixels), are extracted from each strip. The total number of blocks per strip is $n = M \times N$. The vectorized form of the i^{th} block of the j^{th} strip, namely $\boldsymbol{x}_i^j \in \mathbb{R}^m$, is put in i^{th} column of matrix $\boldsymbol{X}^j$, (i.e., $\boldsymbol{X}^j = [\boldsymbol{x}_1^j, \boldsymbol{x}_2^j, \ldots, \boldsymbol{x}_n^j]$).

Then, blocks of the same strip are transformed into the same subspace. As shown in Fig. 10 the j^{th} subspace, S^j, is an eigenspace computed over the j^{th} group of input blocks, i.e., $\boldsymbol{X}^j$, and its corresponding output is obtained by the zero-phase-component analysis (or ZCA-whitening [34]) of j^{th} group of blocks, ($\boldsymbol{X}_W^j = [\boldsymbol{x}_{w1}^j, \boldsymbol{x}_{w2}^j, \ldots, \boldsymbol{x}_{wn}^j]$). As such, the transformation is referred to as "strip-wise ZCA whitening" and its details are given in Algorithm 1. Eigenspaces and eigenvalues of the subspaces (i.e., U^j and D^j, $j = 1, 2, 3, 4$) are learned during the training phase for the local data transformation layer, and are then employed for the test phase.

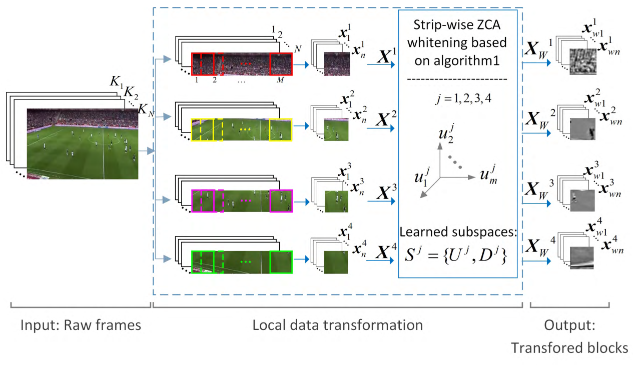


FIGURE 10. Local data transformation component whitens blocks extracted from each strip of input key-frames (X^j), by generating a separate subspace for each strip based on ZCA whitening. Each group of whitened blocks (X_W^j) are passed to the next component for local feature extraction.

Algorithm 1 Strip-Wise ZCA Whitening

```
   Input  : Blocks of 4 strips: $X = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \boldsymbol{X}^3, \boldsymbol{X}^4\}$
   Output: ZCA whitened blocks of 4 strips:
           $X_W = \{\boldsymbol{X}_W^1, \boldsymbol{X}_W^2, \boldsymbol{X}_W^3, \boldsymbol{X}_W^4\}$
1  for strip $j = 1 : 4$ do
2      for block $i = 1 : n$ do
3          Block mean: $\mu_i^j = \frac{1}{m} \sum_{p=1}^{m} x_i^j(p)$
4          Demean the block: $\boldsymbol{x}_i^j = \boldsymbol{x}_i^j - \mu_i^j$
5      end
6      Covariance matrix of blocks of $j^{th}$ strip:
       $\boldsymbol{\Sigma}^j = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i^j (\boldsymbol{x}_i^j)^T$
7      Decompose $\boldsymbol{\Sigma}^j$: $[\boldsymbol{U}^j, \boldsymbol{D}^j, \boldsymbol{V}^j] = \mathrm{SVD}(\boldsymbol{\Sigma}^j)$
8      Save $j^{th}$ subspace: i.e., $S^j = \{\boldsymbol{U}^j, \boldsymbol{D}^j\}$ as learned
       parameter of local data transformater. Where,
       $\boldsymbol{U}^j = [\boldsymbol{u}_1^j, \boldsymbol{u}_2^j, \ldots, \boldsymbol{u}_m^j]$ , $diag(\boldsymbol{D}^j) = \boldsymbol{\lambda}^j$ &
       $\boldsymbol{\lambda}^j = [\lambda_1^j, \lambda_2^j, \ldots, \lambda_m^j]$
9      ZCA Whiten the data:
         • Uncorrelating = Rotating: $\boldsymbol{X}^j\_\mathrm{rot} = (\boldsymbol{U}^j)^T \boldsymbol{X}^j$
         • Rescale data to have identity covariance:
           $\boldsymbol{X}^j\_\mathrm{PCAwhite} = \boldsymbol{X}^j\_\mathrm{rot} / \sqrt{(\lambda^j + \epsilon)}$
         • Rotate back the data:
           $\boldsymbol{X}^j\_\mathrm{ZCAwhite} = \boldsymbol{U}^j \boldsymbol{X}^j\_\mathrm{PCAwhite}$
       Whitened data: $\boldsymbol{X}_W^j = \boldsymbol{X}^j\_\mathrm{ZCAwhite}$
10 end
```

B. PARALLEL STACKED AUTOENCODERS

The second component of PFF-Net, as illustrated in Fig. 5, is composed of four parallel replicas of stacked autoencoders.

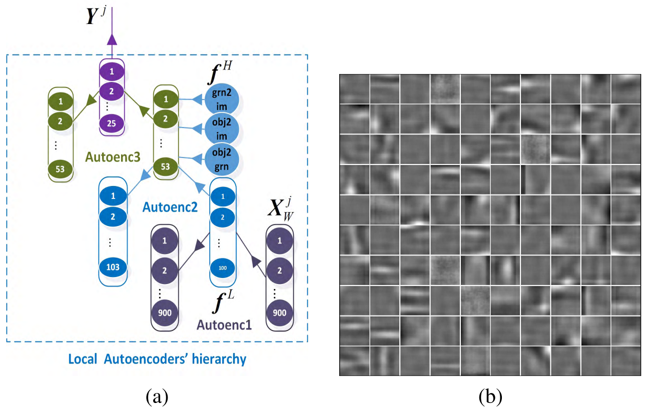


FIGURE 11. (a) A single replica of the stacked autoencoder hierarchy, used in parallel structure of the PFF-Net, that combines the handcrafted features of a key-frame with local features extracted by SAEs from image blocks and provides a lower-dimensional representation of the input blocks. (b) Weights of the first autoencoder in the stacked autoencoder hierarchy which mainly extracts edges from the image blocks.

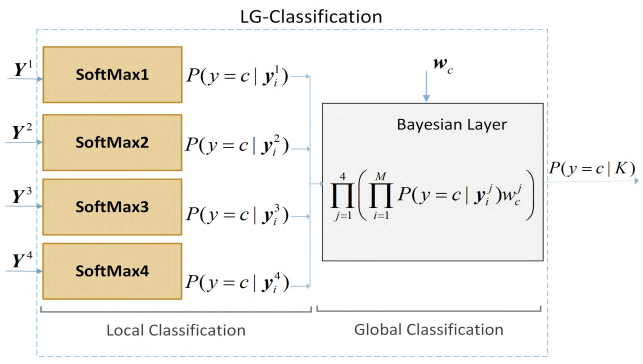
Each single SAE, as shown in Fig. 11(a), models information of a particular strip of key-frames and provides a compressed representation of its input block.

The first autoencoder, in the hierarchy, inputs whitened blocks of size $b \times b = 30 \times 30$ pixels, and generates a 100-dimensional representation, called the local feature vector, $\boldsymbol{f}^L$. Weights of this layer generally extract edges in image (Fig. 11(b)). Local features, $\boldsymbol{f}^L$, are concatenated with the three handcrafted features, $\boldsymbol{f}^H$, describing a whole key-frame, to form a 103-dimensional feature vector, which is passed through the second and the third autoencoders with 53 and 25 hidden neurons, respectively. So, a lower dimensional (25-dimensional) feature vector is achieved that combines local information of the block with the global information of the whole key-frame ($\boldsymbol{y}_i^j \in \mathbb{R}^{25}$ and $\boldsymbol{Y}^j = [\boldsymbol{y}_1^j, \boldsymbol{y}_2^j, \ldots, \boldsymbol{y}_i^j, \ldots, \boldsymbol{y}_n^j]$).

Including the handcrafted features in the hierarchy, causes the network weights to be adjusted not only based on information of a strip, but also according to a bigger picture of an entire frame. Handcrafted features are consistent for all the blocks of the same key-frame and can guide the network in recognizing a bond between the blocks. As shown in Fig. 11(b), in the first autoencoder, edges are detected as local features, so inserting handcrafted features at this layer leads to loss of integrity in the feature structure. Furthermore, empirically by inserting the handcrafted features to different layers of the hierarchy, the second layer is found to be the best location, as the highest classification performance is achieved.

C. LG-CLASSIFICATION

The last component of the network, as shown in Fig. 12, is composed of a two level classification: local and global. The four softmaxes perform local classification. The low-dimensional representation of blocks of jth strip ($\boldsymbol{Y}^j$) achieved



LG-Classification
Y^1
Y^2
Y^3
Y^4
SoftMax1
SoftMax2
SoftMax3
SoftMax4
$P(y=c\mid \boldsymbol{y}_i^1)$
$P(y=c\mid \boldsymbol{y}_i^2)$
$P(y=c\mid \boldsymbol{y}_i^3)$
$P(y=c\mid \boldsymbol{y}_i^4)$
$\boldsymbol{w}_c$
Bayesian Layer
$\prod_{j=1}^{4}\left(\prod_{i=1}^{M} P(y=c\mid \boldsymbol{y}_i^j) w_c^j\right)$
$P(y=c\mid K)$
Local Classification
Global Classification

FIGURE 12. LG-classification component comprises four local softmaxes and a Bayesian layer. Local decision about class of a block, i.e., $P(y = c|\boldsymbol{y}_i^j)$, is made by softmaxes and global decision about class of the whole key-frame, i.e., $P(y = c|K)$, is made by fusing local decisions via the Bayesian layer.

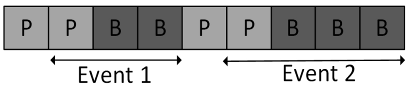


FIGURE 13. A sequence of play/brake labeled shots. Each event span is from last play shot until the last break shot.

from the jth SAE in the PFF-Net is fed into the jth softmax classifier. The local softmaxes determine the probability of each block belonging to each shot view, i.e., $p(y = c|\boldsymbol{y}_i^j, \Theta^j), c \in \{1, 2, 3, 4\}$, represented more concisely as $p(y = c|\boldsymbol{y}_i^j)$.

Afterwards, as shown in Fig. 12, the global classification of a whole key-frame is performed by means of a Bayesian layer. This layer, as given in Eq. 8, multiplies the weighted posterior probabilities assigned to all blocks of all the strips of the input key-frame, K, to achieve the class probability for the input key-frame, $p(y = c|K)$.

$$p(y = c|K) \propto \prod_{j=1}^{4}\left(\prod_{i=1}^{M} p(y = c|\boldsymbol{y}_i^j) w_c^j\right) \tag{8}$$

Details of deriving Eq. 8 is provided in the Appendix. Note that, weights (w_c^j) can be removed from Eq. 8. Finally, the selected class (shot view) of the input key-frame is the one which maximizes $p(y = c|K)$:

$$c^* = \underset{c}{\operatorname{argmax}} p(y = c|K) \tag{9}$$

VI. H2O-MM FOR PLAY-BREAK SEGMENTATION

Play/break labeling of shots in a soccer video can be used for detecting event boundaries. The most effective scope for a soccer event is from the last play shot until the last break shot [1] (Fig. 13). Determining the play/break state of a shot requires the temporal information about the neighboring shots. But, including these information in the PFF-Net is not trivial. It needs the employment of recurrent layers at the top of the network and a big dataset of labeled shots for training. Therefore, here, a separate model is presented for play-break

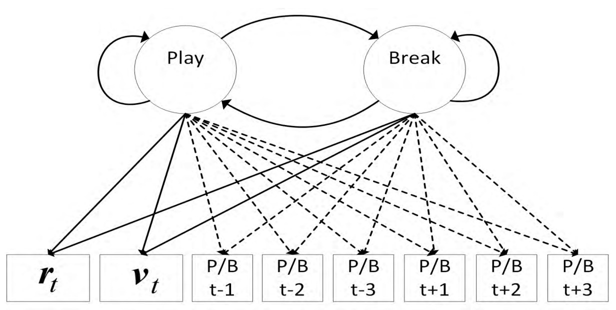


FIGURE 14. Hidden-to-observable transferring Markov Model (H2O-MM) for play/break labeling of shots. Firm links are stable in all iterations. Dashed links are included from the second iteration to incorporate the effect of neighboring shots (three previous shots and three next shots) on state of the current shot.

segmentation which is the Markov model shown in Fig. 14. In this model similar to the HMM that Tavassolopour *et al.* employed [9], shot views and replay label of t^{th} shots ($r_t = 1/0$) are employed as observable variables. The introduced Markov model as shown in Fig. 13 has two hidden states, i.e., play and break, while its observation variables change during the iterative learning and inference processes.

Firm links in Fig. 14 show the connections that exist in all iterations, while dashed links are included after the first iteration and transfer the inferred hidden states in current iteration to the observable variables in the next iteration. As such the presented model is referred to as hidden-to-observable transferring Markov model (H2O-MM).

In each iteration, first transition and emission matrices of the model are estimated, using the labeled data, while 1% noise is added to the ground-truth by randomly altering 1% of the labels from play to break or vice-versa. Adding noise in each iteration, which is inspired from mechanism of exploration in evolutionary algorithms [38], leads to obtaining new transition and emission matrices and is a means for searching the solution space while averting the early convergence to a local solution. Then observation sequence (shot views and replay label of consecutive shots) is given to the model and using the Viterbi algorithm the play/break state of input sequence is inferred.

At the end of each iteration, the inferred hidden states of six neighbors of a shot (i.e., three previous shots and three next shots) are also considered as observable variables for that shot, by adding dashed links in the model. Therefore, state of the current shot is inferred accurately, by iterative refining the output of the existing HMM for play-break segmentation [9], that only uses the past state to infer the current state. The learning and inference mechanism for the H2O-MM is summarized in Algorithm 2.

VII. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, several experiments are conducted to evaluate the performance of the introduced networks for shot view classification and play-break segmentation. In Subsection A, dataset is described and in Subsection B, parameter setting

Algorithm 2 Learning & Inference for H2O-MM

```
Input : 1. Sequence of shot views, and replay labels 2.
        Sequence of play/break labels (ground truth)
Output: Infered play/break sequence
1  if run# = 1 then
2      Consider "Play" & "Break" as hidden states, and
       "View" & "Replay" as observable variable,(i.e.,
       exclude dashed links in Fig. 14). "Ref labels" =
       ground truth.
3  end
4  if run# >= 1 then
5      for itr = 1:max do
6          Estimate Transition & Emission matrices, use
           "Ref labels".
7          Feed observable variables of shots (i.e.,
           observation sequence) to trained H2O-MM, and
           infer play/break label of shots, using Viterbi
           algorithm.
8          "Ref labels" = groundtruth + 1% noise (i.e.,
           alter 1% of labels in ground truth)
9          Transfer inferred hidden states of three
           preceding and three succeeding shots to
           observable variables of the shot (i.e., include
           dashed links in Fig. 14 )
10     end
11 end
```

of SAEs is provided. In Subsections C to E, specifications of the PFF-Net are highlighted, its performance for shot view classification in soccer is compared to related structures and other methods, and its running time per frame is provided. In Subsection F, performance of H2O-MM is compared to two other HMMs used for play-break segmentation in soccer.

A. DATASET

The employed dataset is provided by the Image Processing Laboratory (IPL) at Sharif University. This dataset is composed of 9 hours of 10 soccer videos, collected from several broadcasters, e.g., Spain first division league, England premier league, FIFA world cup, and more. It includes 3452 shots of day-time and night-time matches. The resolution of the frames is 640×340. Two $50 - pixels$ strips from the top and bottom of frames are cut to omit advertisements and frames' resolution is reduced of 320×120 to decrease the computations and memory usage. After several empirical tests with different resolutions, 320×120 was found to be enough for achieving a good shot view classification. However, arbitrary resolutions could be handled by the PFF-Net.

For shot view classification using the PFF-Net, an equal number of shots per class is required. Because in process of driving Eq. 8, the probability of all classes is considered to be the same to simplify the formulas. Therefore, 752 shots (188 shots per class) for the training and 324 shots (81 shots per class) for test are randomly selected from the whole

TABLE 1. Parameter setting for stacked autoencoders' hierarchy.

Parameters	Autoenc1	Autoenc2	Autoenc3
ρ: sparsity proportion	0.1	0.3	0.4
λ: coefficient for L_2 term	0.004	0.002	0.001
β: coefficeint for sparsity	4	4	4
Maximum training epochs	1000	500	500
# of hidden neurons	103 (100)	53 (50)	25

dataset. There are only 269 shots available for out-of-field view, in the dataset. So, for other shot views also the same number of shots is employed. Next, a key-frame has been selected from each shot and $M = 30$ overlapping blocks per strip (with size 30×30 pixels and 10 pixel overlap) are extracted from it. So, 490240 training blocks and 38880 test blocks are generated.

B. PARAMETER SETTING

The parameter setting for the SAE hierarchy of Fig. 11, which is used in different network implementations in Subsection VII.C to VII.E, is given in Table 1. As given in Table 1, the number of hidden neurons decreases through the SAE hierarchy to achieve a compressed representation of input. Forcing sparsity increases the compression further. A high level of compression can lead to significant loss of information which will eventually negatively impact the classification success. To avoid this, the compression ratio $cr^{(l)}$ for each layer l of the SAE is defined in Eq. 10, and is set equal to 90, 6.67 and 5 for the three successive autoencoders. These values are selected empirically by testing different ranges of compression ratios for different layers, in order to achieve a simple network structure which could best discriminate between the classes.

$$cr^{(l)} = \frac{\#\, neurons\ in\ layer\ (l-1)}{(\#\, neurons\ in\ layer\ l) \times \rho^{(l)}} \tag{10}$$

C. EFFECT OF PNs ON PERFORMANCE OF THE PFF-NET

In this experiment, the effect of localized parallel networks PN^j in modelling the data and on performance of the PFF-Net is investigated. For this experiment, the handcrafted feature vector, $\boldsymbol{f}^H$, is excluded from from the PFF-Net, so that the results can be evaluated purely based on the ability of the PNs. Therefore, the architecture of local SAEs is changed from $900-103-53-25$ (neurons in layers) to $900-100-50-25$.

After training, each of the four PNs in the PFF-Net is specialized to locally represent the content of a specific strip. So, $PN^{j'}$ cannot model blocks of an incongruent strip (i.e., $X_W^j, j \neq j'$) as good as PN^j can. To show this, blocks of four strips of test frames are fed into the four parallel networks with different orders, i.e., blocks of j^{th} strip (X_W^j) are fed into j'^{th} parallel network ($PN^{j'}$). The shot views are determined, and correct rates of classification for different data presentation orders are computed and reported in Table 2. To have a fair evaluation, weights are also omitted from Eq. 8, so, our confidence in output of all PNs would be equal, see Appendix. As shown in Table 2, the best result can be achieved when test blocks are delivered to their congruent PNs (i.e., $j = j'$). As an evaluation metric, a measure of distance, namely $\mathcal{O}dist$, is defined (see Eq. 11) where, $\mathcal{O}$ is a set determining delivery order of input data to PNs. The $Odist$ value for different data presentation orders is given in Table 2.

$$\mathcal{O}dist = \sum_{\left\{(j,j')|(X_W^j \rightarrow PN^{j'}) \in \mathcal{O}\right\}} |j - j'| \tag{11}$$

A higher $Odist$ value means blocks of test strips are delivered to PNs that are trained for spatially farther strips. So,performance of the PFF-Net decreases by increase of $Odist$. This experiment shows that the locality in the network, realized by PNs, improves the performance of the network.

D. PERFORMANCE COMPARISON FOR THE PFF-NET

In this experiment, performance of the PFF-Net for shot view classification is compared with the variations of original network, some commonly used classifiers, our implementation of a decision tree [9] with state-of-the-art result, and a CNN-based shot view classifier [30]. Precision, recall and average F1-score are used for comparisons and are given in Table 3. To have a fair comparison, for all of the methods, handcrafted features are computed based on Section IV.

The first two networks in Table 3, namely PFF-Net($W, \boldsymbol{f}^H$) and PFF-Net($!W, \boldsymbol{f}^H$), are variations of the original PFF-Net shown in Fig. 5, where their different parameter settings are provided in parentheses that are included in their names. Sign "!" before a parameter shows that parameter is omitted from the network. So, in both of these networks, handcrafted features, $\boldsymbol{f}^H$, are included, while weight, W, of fusion layer is omitted from the second one.

In the third network, the Bayesian fusion is replaced by a voting fusion. As such, the network is referred to as PFVF-Net($!W, \boldsymbol{f}^H$) (parallel feature with voting fusion network) where handcrafted features are included, but weights of the fusion layer are excluded. In the voting fusion, first the class label for each image block is determined, which is the class with highest posterior probability provided by softmaxes. Next, the number of blocks in each class is counted, and the class with highest vote is assigned to the image. Table 3 shows that the Bayesian fusion outperforms the voting fusion. Giving weight to probabilities improves the results even further.

In the fourth and fifth methods (PFF-Net(W, $!\boldsymbol{f}^H$) and PFVF-Net($!W$, $!\boldsymbol{f}^H$)) handcrafted features are omitted. As a result, performance of PFF-Net drops from 92.6% to 83.3% and performance of PFVF-Net drops from 91.1% to 73.8%. These results show the positive effect of including handcrafted features in networks.

In order to explore the effect of omitting locality from the PFF-Net, a network with a single pipeline (versus the parallel pipelines of PFF-Net) is implemented as the sixth method. As such, the network is referred to as SP-Net($!W$, $!\boldsymbol{f}^H$) (single pipeline network) where both handcrafted features and weights are excluded from the network. In the SP-Net, in the first layer, all blocks are transformed into a common

TABLE 2. Set $\mathcal{O}$ determines order of data presentation to the parallel pipelines of the PFF-Net (i.e., *PN*s). For some different orders of test data input the correct rate of classification is computed. Increase of order distance (i.e., $\mathcal{O}dist$) is equivalent to delivering test strips to a less congruent *PN*, which decreases of classification accuracy.

Presentation order of input to the PFF-Net ($\mathcal{O}$)	Correct Rate	$\lvert j - j' \rvert$	$\mathcal{O}dist$
$\mathcal{O} = \{X_W^1 \rightarrow PN^1, X_W^2 \rightarrow PN^2, X_W^3 \rightarrow PN^3, X_W^4 \rightarrow PN^4\}$	**0.83**	0 0 0 0	0
$\mathcal{O} = \{X_W^1 \rightarrow PN^1, X_W^2 \rightarrow PN^2, X_W^4 \rightarrow PN^3, X_W^3 \rightarrow PN^4\}$	0.79	0 0 1 1	2
$\mathcal{O} = \{X_W^3 \rightarrow PN^1, X_W^2 \rightarrow PN^2, X_W^1 \rightarrow PN^3, X_W^4 \rightarrow PN^4\}$	0.75	2 0 2 0	4
$\mathcal{O} = \{X_W^2 \rightarrow PN^1, X_W^4 \rightarrow PN^2, X_W^1 \rightarrow PN^3, X_W^3 \rightarrow PN^4\}$	0.74	1 2 2 1	6
$\mathcal{O} = \{X_W^3 \rightarrow PN^1, X_W^4 \rightarrow PN^2, X_W^1 \rightarrow PN^3, X_W^2 \rightarrow PN^4\}$	0.68	2 2 2 2	8

TABLE 3. Performance comparison of the PFF-Net for shot view classification with its related architectures (# 1 to # 6) and also with naïve Bayes, softmax classifier, a decision tree and a CNN-based architecture. Different settings of methods are included in parentheses, after their names. Comparison criteria are precision and recall per class, and average F1-score over all classes.

#	Net(*setting*)/Classifier(*setting*)	Criteria	close-up	out-of-field	medium	long	Avg F1-Score
1	PFF-Net(W, $\boldsymbol{f}^H$)	Precision	96.8	92.6	**88.6**	**92.5**	**92.6**
		Recall	96.8	89.3	**88.1**	96.4	
2	PFF-Net($!W$, $\boldsymbol{f}^H$)	Precision	96.8	92.3	86.1	92.5	91.8
		Recall	96.8	85.7	88.1	96.4	
3	PFVF-Net($!W$, $\boldsymbol{f}^H$)	Precision	96.7	**96.0**	83.8	89.4	91.1
		Recall	93.5	85.7	88.1	96.4	
4	PFF-Net(W, $!\boldsymbol{f}^H$)	Precision	89.3	82.1	72.6	92.3	83.3
		Recall	80.6	82.1	83.0	85.7	
5	PFVF-Net($!W$, $!\boldsymbol{f}^H$)	Precision	83.9	82.1	73.7	63.3	73.8
		Recall	83.9	82.1	39.8	**100**	
6	SP-Net($!W$, $!\boldsymbol{f}^H$)	Precision	83.9	78.6	65.6	76.9	76.5
		Recall	83.9	78.6	59.7	85.7	
7	Naïve Bayesian (With Kernel estimation) [39]	Precision	90.0	88.9	78.9	89.3	86.5
		Recall	83.9	78.6	59.7	85.7	
8	Softmax Regression ($\#epochs = 500$) [34]	Precision	90.6	92.6	75.9	80.0	84.8
		Recall	93.5	89.3	71.6	85.7	
9	Decision Tree ($T_{high} = 0.19$, $T_{low} = 0.01$) [9]	Precision	**96.9**	76.9	92.9	87.5	88.0
		Recall	**100.0**	**92.9**	83.3	75.0	
10	CNN [30]	Precision	75.8824		78.5714		77.1
		Recall	79.6296		74.6914		

eigenspace; in the second layer, the four parallel networks are reduced to just one and finally the Bayesian fusion is used for classification. Table 3 shows that there is a huge drop of performance for the SP-Net($!W$, $!\boldsymbol{f}^H$) with respect to the PFF-Net($!W$, $!\boldsymbol{f}^H$), which occurs due to the suppression of locality from the network.

Since a variation of naïve Bayes and softmax regression are utilized as basic elements in the last component of the PFF-Net, these two classifiers are used as the seventh and eighth methods for performance comparison. For the softmax classifier [34], the maximum number of training epochs is set to 500 and in naïve Bayes classifier [39] kernel estimation is performed to achieve the best classification results.

Most of the shot view classifications in the literature are based on a decision tree [9], [12], [16]. Therefore, here a decision tree with state-of-the-art result [9] is also implemented and used as the ninth method for comparison, where the threshold values for it, i.e., $T_{low} = 0.01$ and $T_{high} = 0.19$, are set empirically to achieve the best result. Handcrafted features are used as input for methods seven to nine.

Tabel 3 shows that the implemented decision tree outperforms the softmax and naïve Bayes classifiers, but is still inferior to the three first methods (i.e., variations of the PFF-Net that employ handcrafted features). Precision and recall per class are also generally higher for these three networks, esp. for the PFF-Net(W,$\boldsymbol{f}^H$), in comparison to the other methods.

The tenth method uses a six-layer convolutional neural network [30] which is designed for classification of basketball shot views into close-up and long shots. This network is trained on basketball as well as soccer, ice hockey, volleyball, and rugby video frames, and provides a 4096-dimensional feature vector before its last layer of softmax binary classifier. We tested the trained network on soccer frames. The best performance is achieved when the close-up and out-of-filed views are merged into one class, and medium and long views into another class. Results show the PFF-Net (even without incorporating the handcrafted features) outperforms this binary cnn architecture for shot view classification.

For the first nine methods given in Table 3, F1-score per class is also computed and demonstrated in bar-graphs of Fig. 15. Generally, for bar-graphs in all of the four shot views, the three left-most bars have consistently a higher level. These bars demonstrate the performance of PFF-Net(W,$\boldsymbol{f}^H$), PFF-Net($!W$,$\boldsymbol{f}^H$), and PFVF-Net($!W$,$\boldsymbol{f}^H$) where in all of them handcrafted features and automatically extracted local features are included. F1-Scores for the three right-most methods, namely naïve Bayes, softmax and

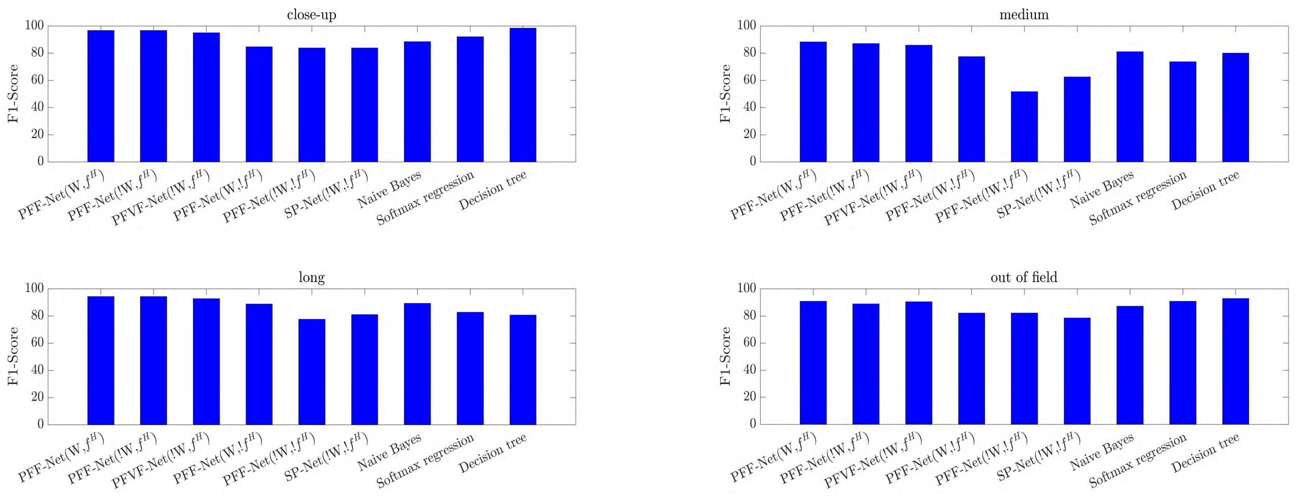


FIGURE 15. F1-score per class (or per shot view), for the first nine methods given in Table 3. Horizontal axis shows the method name and vertical axis is the F1-score for test data. The three left bars have consistently higher level of F1-score for all classes. These three bars demonstrate the performance of some variations of the PFF-Net that employ both handcrafted features and automatically extracted local features.

decision tree, that exclusively employ handcrafted features, stand in the second position. Finally, the three middle bars show the performance of networks that exclude handcrafted features, (i.e., PFF-Net(W, !f^H), PFVF-Net(!W, !f^H), and SP-Net(!W, !f^H)), and in average have the lowest F1-score.

Overall, the PFF-Net(W,f^H) gives the best performance for shot view classification for the given dataset. This is achieved by combining the handcrafted features and automatically extracted local features in the parallel piplines of the network and fusion of weighted probabilities by Bayesian rule.

E. EXECUTION TIME

In this section, run time of introduced networks for shot view classification, explicitly variations of PFF-Net described in Section VII.D, is provided. The reported times given in Table 4 are obtained by averaging the run time per frame over 100 runs. Different networks with their setting are given in the first columns of Table 4. Columns two to four respectively show the average run time/frame for extracting handcrafted features, passing the frame through the trained network, and finally, the total run time/frame for shot view classification. Therefore, shot view classification for a complete soccer game (90 minutes long) roughly takes 2 minutes using the PFF-Net.

All the reported times are measured on a desktop computer Intel Core i5- 2400s @ 2.50GHz with 6GB RAM, using Matlab 2016a. It is notable that in the training phase of the PFF-Net, strip-wise ZCA whitening is performed with time complexity of $O(m^2n)$, and each of the four *PNs* in the PFF-Net is trained and fine-tuned separately on a set of n blocks, extracted from a strip of training images.

F. H2O-MM PLAY-BREAK EVALUATION

Here, the accuracy of play-break segmentation using H2O-MM on the dataset described in Section VII.A

TABLE 4. Average run time per frame for Shot view classification using different network architectures based on PFF-Net. Columns 2 to 4 respectively show run time for: Handcrafted feature extraction, passing through the network, and total run time.

Network structure	Avg Time/frame (sec)		
Net($Settings$)	f^H Extraction	Net pass	Total
PFF-Net(W, f^H)	0.0880	0.1210	0.2090
PFF-Net(!W, f^H)	0.0880	0.1193	0.2073
PFVF-Net(!W, f^H)	0.0880	0.1210	0.2090
PFF-Net(W, !f^H)	0	0.1259	0.1259
PFVF-Ne(!W, !f^H)	0	0.1251	0.1251
SP-Net(!W, !f^H)	0	0.1153	0.1153

TABLE 5. Play-break segmentation accuracy using various HMMs.

Method :	HMM [11]	HMM [9]	H2O-MM
Accuracy	83.5%	97%	**98.7%**

is computed and compared to two existing hidden Markov models developed for this purpose [9], [11] and results are presented in Table 5. Xie *et al.* [11] presented an HMM that employs dominant color ratio, and motion intensity of frames as observation variables, while Tavassolipour *et al.* [9] employ shot views and replay labels as observation variables and achieve the-state-of-the-art result for play-break segmentation.

In the introduced hidden Markov model, namely H2O-MM (see Section VI), the maximum number of iterations in Algorithm 2 is set to 100. The algorithm has been run 10 times and the average accuracy over all iterations in the last run is reported as the obtained accuracy. As given in Table 5, the accuracy of H2O-MM is higher than the two existing HMMs.

VIII. CONCLUSION

In this article, we have designed and implemented a parallel deep network based on stacked autoencoders (PFF-Net) for soccer shot view classification and a hidden Markov model (H2O-MM) for play-break segmentation of broadcast soccer videos, that can be employed for higher level analysis of a soccer game.

The PFF-Net, using parallel SAEs, effectively models information of distinct strips of soccer key-frames and combines them with handcrafted features of the full scene to produce lower-dimensional representations for image blocks. Next, the generated representations are passed through softmaxes and a fussing Bayesian layer to classify the shot views.

The shot views are then given to the H2O-MM, which has a novel learning and inference mechanism that transfers hidden states to observation variables. Experiments show the H2O-MM gives the highest accuracy in play-break segmentation of soccer video compared to the best reported results.

Although the PFF-Net is purposefully designed for soccer shot view classification, it could inspire networks for other applications where a local similarity could be found in data. The new mechanism employed in H2O-MM also could be utilized in other HMMs to increase the accuracy of inference.

APPENDIX

To derive Eq. 8 of Section V.C we start with assumption made in a naïve Bayesian classifier, where attributes (features), i.e., A_i, $i = 1, 2, \ldots, n$, are assumed to be conditionally independent given the class value y. So, we can write

$$P(A_1, A_2, \ldots, A_n, y) = P(y) \prod_{i=1}^{n} P(A_i|y) \tag{12}$$

Having the naïve independence assumption, we are interested in expressing $P(y|A_1, A_2, \ldots, A_n)$ based on $P(y|A_i)$. By using the Bayesian rule, we have

$$\begin{aligned} P(A_1, A_2, \ldots, A_n, y) &= P(y) \prod_{i=1}^{n} P(A_i|y) \\ &= P(y) \prod_{i=1}^{n} \frac{P(y|A_i)P(A_i)}{P(y)} \end{aligned} \tag{13}$$

$$\begin{aligned} P(y|A_1, A_2, \ldots, A_n) &= \frac{P(y)}{P(A_1, A_2, \ldots, A_n)} \\ \prod_{i=1}^{n} \frac{P(y|A_i)P(A_i)}{P(y)} &\propto \frac{1}{P(y)^{n-1}} \prod_{i=1}^{n} P(y|A_i) \end{aligned} \tag{14}$$

In Eq. 14, probabilities that are not related to the class variable are omitted. Considering equal probabilities for all classes (see Section VII.A), we get

$$\begin{aligned} c^* &= \underset{c}{\text{argmax}} P(y = c|A_1, A_2, \ldots, An) \\ &= \underset{c}{\text{argmax}} \prod_{i=1}^{n} P(y = c|A_i = a_i) \end{aligned} \tag{15}$$

In the Bayesian layer of the PFF-Net, we are interested to determine the class label that maximizes $P(y = c|K)$. We assume that, $P(y|K) = P(y|\mathbf{y}_1^1, \ldots, \mathbf{y}_n^1, \mathbf{y}_1^2 \ldots, \mathbf{y}_n^2, \ldots, \mathbf{y}_1^4 \ldots, \mathbf{y}_n^4)$, which means probability of a class given a frame is equal to probability of a class given lower-dimensional representations of all blocks of all the four strips of that frame. Therefore, if we consider features $\mathbf{y}_i^j$s to be conditionally independent given the class value y, based on Eq. 14 we have

$$c^* = \underset{c}{\text{argmax}} \prod_{j=1}^{4} \left(\prod_{i=1}^{n} P(y = c|\mathbf{y}_i^j) \right) \tag{16}$$

Note that in Eq. 8, the conditional probabilities of Eq. 16 are replaced by their weighted versions, i.e., $P(y = c|\mathbf{y}_i^j)w_c^j$. Weights are determined using normalized similarity matrices, $\boldsymbol{M}_{simc}$, defined in Section V.A. Vector $\boldsymbol{w}_c = [w_c^1, w_c^2, w_c^3, w_c^4]$ is equal to $diag(\boldsymbol{M}_{simc})$. The weight w_c^j which is associated to the posterior probability of the PN^j for class c, is able to determine the level of confidence in that probability. Because each PN models information of blocks of an specific strip. So, a PN that learns a group of similar blocks can provide a better model of those blocks compared to one that learns a group of diverse blocks. Therefore, confidence in the former should be higher than the latter.

ACKNOWLEDGMENT

The authors would like to thank Dr. S. Kasayee and her team in Image Processing Laboratory of Sharif University who let us have access to their soccer video dataset.

REFERENCES

[1] D. W. Tjondronegoro and Y.-P. P. Chen. "Knowledge-discounted event detection in sports video," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 5, pp. 1009–1024, Sep. 2010.

[2] X. Qian, H. Wang, G. Liu, and X. Hou, "HMM based soccer video event detection using enhanced mid-level semantic," *Multimedia Tools Appl.*, vol. 60, no. 1, pp. 233–255, 2012. [Online]. Available: https://doi.org/10.1007/s11042-011-0817-y

[3] M. Archana and M. K. Geetha, "Object detection and tracking based on trajectory in broadcast tennis video," *Procedia Comput. Sci.*, vol. 58, pp. 225–232, Aug. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915021717

[4] P. Oskouie, S. Alipour, and A.-M. Eftekhari-Moghadam, "Multimodal feature extraction and fusion for semantic mining of soccer video: A survey," *Artif. Intell. Rev.*, vol. 42, no. 2, pp. 173–210, Aug. 2014. [Online]. Available: https://doi.org/10.1007/s10462-012-9332-4

[5] C. Xu, J. Cheng, Y. Zhang, Y. Zhang, and H. Lu, "Sports video analysis: Semantics extraction, editorial content creation and adaptation," *J. Multimedia*, vol. 4, no. 2, pp. 69–79, 2009.

[6] D. A. Sadlier and N. E. O'Connor, "Event detection in field sports video using audio-visual features and a support vector machine," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1225–1233, Oct. 2005.

[7] J. Assfalg, M. Bertini, C. Colombo, and A. D. Bimbo, "Semantic annotation of sports videos," *IEEE MultimediaMag.*, vol. 9, no. 2, pp. 52–60, Apr. 2002.

[8] J. Liu, X. Tong, W. Li, T. Wang, Y. Zhang, and H. Wang, "Automatic player detection, labeling and tracking in broadcast soccer video," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 103–113, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865508000627

[9] M. Tavassolipour, M. Karimian, and S. Kasaei, "Event detection and summarization in soccer videos using Bayesian network and Copula," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 2, pp. 291–304, Feb. 2014.

[10] T. D'Orazio *et al.*, "An investigation into the feasibility of real-time soccer offside detection from a multiple camera system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 12, pp. 1804–1818, Dec. 2009.

[11] L. Xie, P. Xu, S.-F. Chang, A. Divakaran, and H. Sun, "Structure analysis of soccer video with domain knowledge and hidden Markov models," *Pattern Recognit. Lett.*, vol. 25, no. 7, pp. 767–775, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865504000145

[12] A. Ekin, A. M. Tekalp, and R. Mehrotra, "Automatic soccer video analysis and summarization," *IEEE Trans. Image Process.*, vol. 12, no. 7, pp. 796–807, Jul. 2003.

[13] A. Rehman and T. Saba, "Features extraction for soccer video semantic analysis: Current achievements and remaining issues," *Artif. Intell. Rev.*, vol. 41, no. 3, pp. 451–461, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1007/s10462-012-9319-1

[14] C.-L. Huang, H.-C. Shih, and C.-Y. Chao, "Semantic analysis of soccer video using dynamic Bayesian network," *IEEE Trans. Multimedia*, vol. 8, no. 4, pp. 749–760, Aug. 2006.

[15] P. P. Mohanta, S. K. Saha, and B. Chanda, "A model-based shot boundary detection technique using frame transition parameters," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 223–233, Feb. 2012.

[16] X. Tong, Q. Liu, and H. Lu, "Shot classification in broadcast soccer video," *ELCVIA Electron. Lett. Comput. Vis. Image Anal.*, vol. 7, no. 1, pp. 16–25, 2008. [Online]. Available: http://elcvia.cvc.uab.es/article/view/270

[17] M. Yazdi and M. Fani, "Shot boundary detection with effective prediction of transitions' positions and spans by use of classifiers and adaptive thresholds," in *Proc. 24th Iranian Conf. Electr. Eng. (ICEE)*, May 2016, pp. 167–172.

[18] Z.-M. Lu and Y. Shi, "Fast video shot boundary detection based on SVD and pattern matching," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 5136–5145, Dec. 2013.

[19] G. Pal, D. Rudrapaul, S. Acharjee, R. Ray, S. Chakraborty, and N. Dey, *Video Shot Boundary Detection: A Review*. Cham, Switzerland: Springer, 2015, pp. 119–127. [Online]. Available: https://doi.org/10.1007/978-3-319-13731-5 14

[20] A. M. Ferman and A. M. Tekalp, "Fuzzy framework for unsupervised video content characterization and shot classification," *J. Electron. Imag.*, vol. 10, no. 4, pp. 917–929, 2001. [Online]. Available: http://dx.doi.org/10.1117/1.1406946

[21] M. H. Sigari, H. Soltanianzadeh, and H. R. Pourreza, "Fast highlight detection and scoring for broadcast soccer video summarization using on-demand feature extraction and fuzzy inference," *Int. J. Comput. Graph.*, vol. 6, no. 1, pp. 13–36, Jan. 2015.

[22] M.-H. Sigari, H. Soltanian-Zadeh, V. Kiani, and A.-R. Pourreza, "Counterattack detection in broadcast soccer videos using camera motion estimation," in *Proc. Int. Symp. Artif. Intell. Signal Process. (AISP)*, Mar. 2015, pp. 101–106.

[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. Hoboken, NJ, USA: Wiley, 2000.

[24] Y. LeCun, Y. Bengio, and G. Hinton, "Review: Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[25] D. Tjondronegoro, Y.-P. P. Chen, and B. Pham, "The power of play-break for automatic detection and browsing of self-consumable sport video highlights," in *Proc. 6th ACM SIGMM Int. Workshop Multimedia Inf. Retr. (MIR)*, 2004, pp. 267–274. [Online]. Available: http://doi.acm.org/10.1145/1026711.1026755

[26] S. Pouyanfar and S.-C. Chen, "Semantic event detection using ensemble deep learning," in *Proc. IEEE Int. Symp. Multimedia (ISM)*, Dec. 2016, pp. 203–208.

[27] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proc. IEEE Int. Symp. Circuits Syst.*, May/Jun. 2010, pp. 253–256.

[28] N. Neverova, P. Luc, C. Couprie, J. J. Verbeek, and Y. LeCun, "Predicting deeper into the future of semantic segmentation," in *Proc. ICCV*, 2017, pp. 648–657. [Online]. Available: http://arxiv.org/abs/1703.07684

[29] M. Wagenaar, E. Okafor, W. Frencken, and M. A. Wiering, "Using deep convolutional neural networks to predict goal-scoring opportunities in soccer," in *Proc. ICPRAM*, 2017, pp. 448–455.

[30] J.-H. Park and K. Cho, "Extraction of visual information in basketball broadcasting video for event segmentation system," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2016, pp. 1098–1100.

[31] S. Gerke, K. Müller, and R. Schäfer, "Soccer jersey number recognition using convolutional neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 734–741.

[32] C. V. Sheena and N. K. Narayanan, "Key-frame extraction by analysis of histograms of video frames using statistical methods," *Procedia Comput. Sci.*, vol. 70, pp. 36–40, Nov. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915031853

[33] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Boston, MA, USA: Addison-Wesley, 2001.

[34] A. Ng, J. Ngiam, C. Y. Foo, Y. Mai, and C. Suen. (2010). *UFLDL Tutorial*. [Online]. Available: http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial

[35] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[36] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, 1997. Online]. Available: http://www.sciencedirect.com/science/article/pii/S0042698997001697

[37] M. Fani, N. Pourjafarian, E. Taherianfard, and M. Yazdi, "Log-spectrum based RSTB invariant template matching with modified ICA," in *Proc. 5th Int. Symp. Telecommun.*, Dec. 2010, pp. 787–792.

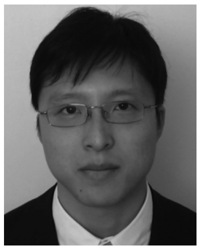
[38] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 45, no. 3, pp. 35:1–35:33, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2480741.2480752

[39] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell. (UAI)*, San Francisco, CA, USA, 1995, pp. 338–345. [Online]. Available: http://dl.acm.org/citation.cfm?id=2074158.2074196

MEHRNAZ FANI (M'09) received the B.A.Sc. degree in electronics engineering from the Shiraz University of Technology, Shiraz, Iran, in 2006, and the M.Sc. degree in electronics and communications engineering, Shiraz University, Shiraz, in 2009, where she is currently pursuing the Ph.D. degree in electronics and communications engineering. Her primary research interests include sport video content analysis using machine learning algorithms. Her research interests include video/image processing, statistical signal processing, pattern recognition, machine learning, and computer vision algorithms, with a specific emphasis on probabilistic models and deep networks.

MEHRAN YAZDI (M'10) received the B.Sc. degree in communication systems from the Department of Electrical Engineering, Shiraz University, Shiraz, Iran, in 1992, and the M.Sc. and Ph.D. degrees in digital vision and image processing from the Department of Electrical Engineering, Laval University, Quebec, QC, Canada, in 1996 and 2003, respectively. He is currently an Associate Professor with the Department of Communications and Electronics Engineering, Shiraz University. He conducted several projects in the area of hyperspectral image compression and denoising, CT metal artifact reduction, and video compression. His major research interests are in the field of image/video processing, remote sensing, multidimensional signal processing, and medical image analysis.

DAVID A. CLAUSI (S'93–M'96–SM'03) received the Ph.D. degree in systems design engineering from the University of Waterloo, Waterloo, ON, Canada, in 1996. He was involved in medical imaging with Agfa, Waterloo. He started his academic career as an Assistant Professor of geomatics engineering with the University of Calgary, Canada, in 1997. In 1999, he returned to his Alma Mater and is currently a Professor, specializing in the fields of intelligent and environmental systems. He was the Associate Chair of the graduate studies from 2008 to 2012. He is also an active interdisciplinary and multidisciplinary Researcher. He has an extensive publication record, publishing refereed journal and conference papers in the diverse fields of remote sensing, computer vision, algorithm design, and biomechanics. His research efforts have led to successful commercial implementations, including creating, building, and selling a company. He has received numerous scholarships, paper awards, research awards, and two teaching excellence awards. In 2010, he received the award for research excellence and service to the research community from the Canadian Image Processing and Pattern Recognition Society. He was the Co-Chair of the IAPR Technical Committee 7—Remote Sensing from 2004 to 2006.



ALEXANDER WONG (M'05–SM'16) received the B.A.Sc. degree in computer engineering, the M.A.Sc. degree in electrical and computer engineering, and the Ph.D. degree in systems design engineering from the University of Waterloo, Waterloo, ON, Canada, in 2005, 2007, and 2010, respectively. He is currently the Canada Research Chair of Medical Imaging Systems, the Co-Director of the Vision and Image Processing Research Group, and an Associate Professor with the Department of Systems Design Engineering, University of Waterloo. He has authored over 400 refereed journal and conference papers, and patents in various fields, such as computational imaging, artificial intelligence, computer vision, medical imaging and analysis, and multimedia systems. He has received numerous awards, including two Outstanding Performance Awards, the Distinguished Performance Award, the Engineering Research Excellence Award, the Sandford Fleming Teaching Excellence Award, the Early Researcher Award from the Ministry of Research and Innovation, and over ten paper awards from national and international conferences, and the Alumni Gold Medal.