STRUCTURE ANALYSIS OF SOCCER VIDEO WITH HIDDEN MARKOV MODELS

Lexing Xie, Shih-Fu Chang
Department of Electrical Engineering
Columbia University, New York, NY
{xlx, sfchang}@ee.columbia.edu

Ajay Divakaran, Huifang Sun
Mitsubishi Electric Research Lab
Murray Hill, NJ
{ajayd, hsun}@merl.com

ABSTRACT

In this paper, we present algorithms for parsing the structure of produced soccer programs. The problem is important in the context of a personalized video streaming and browsing system. While prior work focuses on the detection of special events such as goals or corner kicks, this paper is concerned with generic structural elements of the game. We begin by defining two mutually exclusive states of the game, play and break based on the rules of soccer. We select a domain-tuned feature set, dominant color ratio and motion intensity, based on the special syntax and content characteristics of soccer videos. Each state of the game has a stochastic structure that is modeled with a set of hidden Markov models. Finally, standard dynamic programming techniques are used to obtain the maximum likelihood segmentation of the game into the two states. The system works well, with 83.5% classification accuracy and good boundary timing from extensive tests over diverse data sets.

1. INTRODUCTION

In this paper, we present new algorithms for soccer video structure analysis. The problem is useful in automatic content filtering for soccer fans and professionals, and it is more interesting in the broader background of video structure analysis and content understanding. By structure, we are primarily concerned with the temporal sequence of high-level game states, namely *play* and *break*, and the goal of this paper is to parse the continuous video stream into an alternating sequence of the two states automatically. This approach is distinctive from existing works, most of which focus on the detection of domain-specific events. And the advantages of parsing structures separately from event detection are: (1) typically no more than 60% of content corresponds to *play*, thus we can achieve significant information reduction; (2) content characteristics in *play* and *break* are different, thus we can optimize event detectors with such prior knowledge.

Related work in the literature mainly lies in sports video analysis, including soccer and various other games, and general video segmentation. For soccer video, prior work has been on shot classification [2], scene reconstruction [8], and rule-based semantic classification [6]. For other sports video, supervised learning was used in [9] to recognize canonical views such as baseball pitching and tennis serve. For general video classification, hidden Markov models (HMM) is used [3] to distinguish different types of programs such as news, commercial, etc. Our previous work [7] built heuristic rules using a domain-specific feature, dominant color ratio, to segment *play* and *break*. The work presented in this paper focuses on two specific aspects that were not investigated in the previous work: (1) using formal statistical techniques to model domain-specific syntactic constraints rather than constructing heuristic rules directly; (2) using simple, but effective features to capture the content syntax.

We first define *play* and *break* as the set of soccer semantic alphabets used in this paper, and then we select two features based on observations of soccer video syntax: dominant color ratio and motion intensity. The stochastic structure within a *play* or a *break* is modeled with a set of HMMs, and the transition among these HMMs is captured with dynamic programming. Average classification accuracy per segment is above 80%, and most of the *play/break* boundaries are correctly detected within a 3-second offset.

Section 2 presents relevant observations of soccer video syntax and the selection of features; section 3 includes algorithms for HMM training and classification; section 4 describes our experiments and results in greater detail; section 5 concludes the paper.

2. VIDEO SYNTAX AND FEATURE SELECTION

2.1 Soccer game semantics

We define the set of mutually exclusive and complete semantic states in a soccer game: *play* and *break* [5]. The game is *in play* when the ball is in the field and the game is going on; *break*, or *out of play*, is the compliment set, i.e. whenever "the ball has completely crossed the goal line or touch line, whether on the ground or in the air" or "the game has been halted by the referee".

Segmenting a soccer video into *play /break* is hard because of: (1) the absence of a canonical scene (such as the serve scene in tennis or the pitch scene in baseball video [9]); (2) the loose temporal structure, i.e. *play/break* transitions and highlights of a game (goal, corner kick, shot, etc) do not have a deterministic relationship with other perceivable events (as opposed to volleys are always preceded by a serve in a tennis game). Yet identifying *play*/*break* is interesting because not only can we achieve about 40% information reduction (Table 1), *play*/*break* information also has potential applications such as play-by-play browsing and editing, or play-break game statistics analysis.

2.2 Soccer video syntax

Soccer video syntax refers to the typical production style and editing patterns that help the viewer understand and appreciate the game. Two major factors influencing the syntax are the producer and the game itself, and the purpose of syntax is to emphasize the events as well as to attract viewers' attention (such as the use of cutaways). Specifically, soccer video syntax can be characterized by some rules-of-thumb observed by sports video producers [1]: (1) convey global status of the game; (2) closely follow action and capture highlights. In our algorithm, two salient features are selected to capture this syntax implicitly.

2.3 Feature extraction

Dominant-color ratio:

As shown in [7], the color of grass field can be adaptively learned for each clip by picking up the dominant hue value throughout a randomly selected frame pool. Hence we can distinguish *grass* pixels vs. *non-grass* pixels in each frame. Define dominant-color-ratio as:

$$\eta_c = |P_g| / |P|, \quad (2.1)$$

where P is the set of pixels, and P_g is the set of *grass* pixels.

Observations in [7] also showed that η_c indicates the scale of view in the current shot, which falls into one of the following three categories: the *wide* (or *global*) shot, with the largest percentage of grass field; the *medium* (or *zoom-in* shot), with less grass in sight; and the *close-up* (including cutaways), with few grass pixels. Moreover, as consistent with the production principles mentioned in the previous section, a *play* is usually captured by *wide* shots interleaved with short *medium* shots or *close-ups*; and a *break* usually has a majority of *close-up* and *medium* shots. However, we are analyzing features uniformly sampled from the video stream rather than the key-frame of each shot [2], because: (1) shots are neither aligned with the *play/break* states nor consistent with the scale of view; (2) shot detectors tend to give lots of false alarms due to unpredictable camera motion and intense object motion.

The dominant-color ratio was thresholded in [7] in order to map it to three types of view scales directly. And in this paper, the dominant-color-ratio η_c is modeled as the Gaussian observations of HMMs described in section 3.1.

Motion intensity:

Motion intensity m is computed as the average magnitude of "effective" motion vectors in a frame (equation 2.2).

$$m = \frac{1}{|\Phi|}\sum_{\Phi}\sqrt{v_x^2 + v_y^2}, \quad (2.2)$$

where $\Phi = \{\text{inter-coded macro-blocks}\}$, and $\vec{v} = [v_x, v_y]$ is the motion vector for each macro-block.

Motion intensity roughly estimates the gross motion in the whole frame, including object and camera motion. It carries complementary information to the color feature, and it often indicates the semantics within a particular shot. For instance, a wide shot with high motion intensity often results from player motion and camera pan during a *play*; while a static wide shot usually occurs when the game has come to a pause.

In the sample clip shown in Figure 1, we can see distinct feature patterns are associated with the scale of shot and the game status. But as these variations are hard to quantify with explicit low-level decision rules, we resort to HMM modeling described in the next section.

3. PLAY-BREAK CLASSIFICATION

In this section, classification algorithms using HMM and dynamic programming are presented.

Soccer game has distinct inherent states *play* (*P*) and *break* (*B*), and each of these two broad classes also consists of different sub-structures such as the switching of shots and the variation of motion. This is analogous to isolated word recognition [4] where models for each word are built and evaluated with the data likelihood. But as these domain-specific classes *P/B* in soccer are very diverse in themselves (typically ranging from 6 seconds up to 2 minutes in length), we use a set of models for each class to capture the structure variations. And this differs from just using a homogeneous model for each class as in [3].

Our task here is to segment and classify a continuous feature stream in one pass. Hence we take a fixed-length sliding window (width 3 seconds, sliding by 1 second) and classify the feature vector into either one of the *P/B* classes. The feature stream is first smoothed by a temporal low-pass filter, normalized with regard to its mean and variance of the entire clip, then the segment of size 2xN in each time slice (2 is feature dimension, N is window length) is fed into the HMM-dynamic programming modules for classification.

3.1 Compute HMM model likelihood

Denote *play* models/likelihoods with subscript *P*, and *break* models/likelihoods with *B* hereafter. Let the set of pre-trained HMM be: $\Omega \hat{=} \Omega_P \bigcup \Omega_B = \{P1,...Pn; B1...Bn\}$, we evaluate the feature vector likelihood under each of the models, to get the set of likelihoods for each time slice, denoted as: $\vec{Q}(t) = [Q_{P1}(t),...Q_{Pn}(t), Q_{B1}(t)...Q_{Bn}(t)]$ (Figure 2, left part).

In our system, 6 HMM topologies are trained for *play* and for *break*, respectively. These include 1/2/3-state fully connected models, 2/3 state left-right models and a 2-state fully connected model with an entering and an exiting state. The observations are modeled as mixture of Gaussians, and we have 2 mixtures per feature dimension per state in the experiments.

The HMM model parameters are trained using the EM algorithm[4]. Training data are manually chopped into homogeneous *play/break* chunks; EM for the *play*-models are conducted over every complete *play* chunks, and vice versa for *break*-models. HMM training is not conducted over 3-second windows because we hope that the HMM structures can take

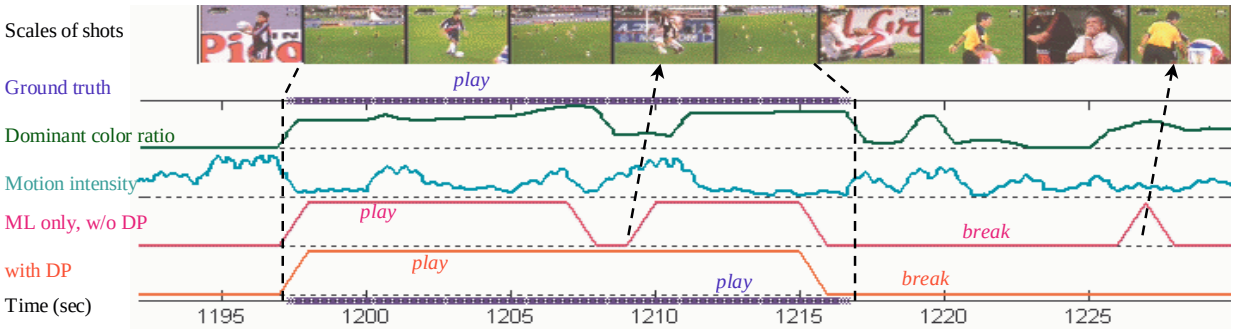

Figure 1. Clip *Argentina* (19'52"~20'30"): key frames, feature contours, and segmentation results (with or without dynamic programming)

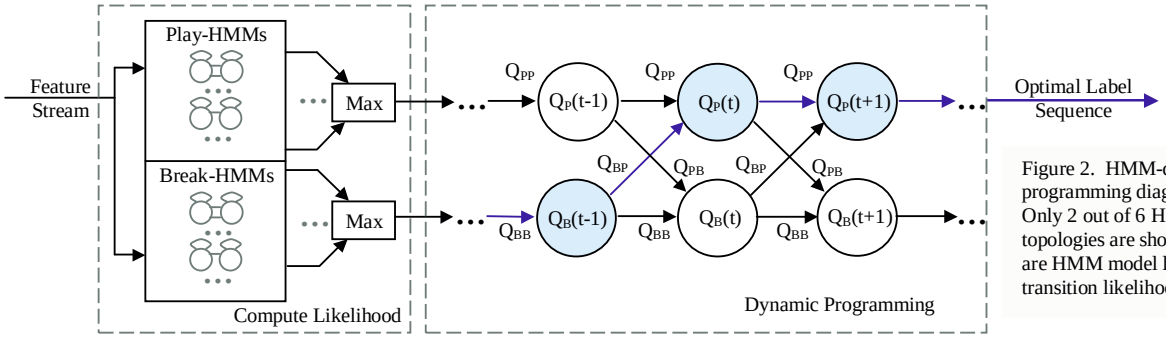

Figure 2. HMM-dynamic programming diagram. Only 2 out of 6 HMM topologies are shown; the *Q*s are HMM model likelihood or transition likelihood

longer time correlation into account, and thus "tolerate" some less frequent events in a state, such as short *close-ups* within a *play*. Experiments show that the overall accuracy will be consistently 2~3% lower if models are trained on short segments, and the video tends to be severely over-segmented as some of the short close-ups and cutaways during a *play* will be misclassified as *break*. Moreover, *Student's t-test* shows that the null hypothesis that training on longer and short segments have the same accuracy is rejected with 95.0% confidence.

Since training is done for the whole play or break, but classification is done over short segments, we may conjecture that results will not be worse if only the three fully connected models (instead of all six) are used. This is confirmed by the result that classification accuracy only differs by 1.5% for these two cases and such a difference is not significant since the *t-test* confidence is less than 50%.

3.2 Find optimal path with dynamic programming

HMM likelihood tells about the "fitness" of each model for every segment, but the long-term correlation is unaccounted for. Thus, finding a global optimal state path $\{s(t) \mid t = 1,2,..T, s(t) = P/B\}$ using neighborhood information is our next step. At each time interval, define 2 nodes corresponding to states *P* and *B*, respectively; the score of each node is the likelihood of the "best-fit" among all 6 models for that state:

$$Q_P(t) = \max_i \{Q_{Pi}(t)\},\ Q_B(t) = \max_i \{Q_{Bi}(t)\},\ i = 1, \ldots, 6.$$

Also define the transition likelihood from one state of the previous interval to a state of the current interval as Q_{PP}, Q_{PB}, Q_{BP}, Q_{BB}, obtained by counting over the training set:

$$Q_{PP} = \log P\{s(t+1) = P \mid s(t) = P\} = \log \sum_{t=1}^{T-1} \frac{\delta_p(t)\delta_p(t+1)}{\delta_p(t)},$$

where $\delta_p(t) = 1$ if *s(t)*=***P***, *0* otherwise.

Similarly, we can define Q_{PB}, Q_{BP} and Q_{BB}.

Hence we have a trellis grid (Figure 2, right) with scores associated with each node and each transition, and dynamic programming[4] is a well-established technique for finding the best path on this grid. Let $\sigma_P(t)$ and $\sigma_B(t)$ be the highest score along a single path that leads to state *P* and *B* at time *t*, respectively, then we can identify the best scores for state *P* and *B* at time *t+1*:

$$\sigma_P(t+1) = (1-\lambda)Q_P(t+1) + \max\{\lambda Q_{PP} + \sigma_P(t),\ \lambda Q_{BP} + \sigma_B(t)\}$$

$$\sigma_B(t+1) = (1-\lambda)Q_B(t+1) + \max\{\lambda Q_{PB} + \sigma_P(t),\ \lambda Q_{BB} + \sigma_B(t)\}$$

Here the transitions are only modeled between play and break, rather than among all of the underlying HMM models, because having this 2x2 transition matrix is sufficient for our play/break segmentation task, and modeling all possible transitions among all HMMs (a 12x12 transition matrix required) is subject to over-fitting. If the score $Q_P(t)$ and $Q_B(t)$ at each node were the true posterior probability that feature vector at time *t* comes from a *play* or a *break* model, then this dynamic programming step would essentially be a second-level HMM. Here constant λ weights model likelihood and transition likelihood: $\lambda = 0$ is equivalent to maximum likelihood classification; $\lambda = 1$ gives a first-order Markov model. Classification accuracy is not very sensitive to λ, if valued within a reasonable range. A typical λ is 0.25, and classification accuracy varies within $\pm 1.5\%$for $\lambda \in [0.1, 0.4]$.

As demonstrated in figure 1, employing this dynamic programming step alleviates over-segmentation, and results show that average classification accuracy is improved by 2.2% over HMM-maximum likelihood only, with *t-test* confidence 99.5%.

4. EXPERIMENTS AND EVALUATION

Four soccer video clips used in our experiment are briefly described in Table 1. All clips are in MPEG-1 format, SIF size, 30f/s or 25f/s; dominant color ratio and motion intensity are computed on I- and P-frames only; motion intensity is interpolated on I-frames. The ground-truth is labeled under the principles that (1) we assume the game status does not change unless indicated by a perceivable event; (2) replays are treated as *in play*, unless it is not adjacent to a play and shorter than 5 seconds. Here *play-percentage* refers to the amount of time the game is in *play* over the total length of the clip.

Clip Name	Length	# of plays	play-percentage	Source
Argentina	23'56"	34	58.5%	TV program
KoreaA	25'00"	37	60.6%	Mpeg-7
KoreaB	25'23"	28	52.1%	Mpeg-7
Espana	15'00"	16	59.2%	Mpeg-7

Table 1. Soccer video clips used in the experiment

In our experiments, the HMM are trained on one clip and tested on other three clips; this process is repeated four times. Our first measurement is the *classification accuracy*, defined as the number of correctly classified segments over total number of segments. Training and testing accuracies are shown in Table 2. Average classification performance (*avg-cla*) of each clip as test set is computed as the mean of the non-diagonal elements of the current row; similarly, average generalization performance

(*avg-gen*) is computed for the clip as training set; and the overall average classification/generalization accuracy over the entire dataset is put in the lower right corner.

Test Set	**Training Set**				**avg-cla**
	Argentina	KoreaA	KoreaB	Espana	
Argentina	0.872	0.825	0.825	0.806	0.819
KoreaA	0.781	0.843	0.843	0.798	0.807
KoreaB	0.799	0.853	0.853	0.896	0.849
Espana	0.799	0.896	0.896	0.817	0.863
avg-gen	0.793	0.858	0.855	0.833	0.835

Table 2. Classification accuracy,
the diagonal elements are training results

Since our goal is to do joint segmentation and classification in one-pass, we are also interested in measuring the boundary accuracy. For each 3-second segment (1 second apart from each other), the classifier not only gives the P/B label, but also indicates if a boundary exists between the previous and the current label. This is different from boundary detection algorithms that solely aim at outlier detection (such as shot boundary detection by measuring histogram distance), since each misjudgment here can cause two false positives instead of one. Therefore, we look at the whole confusion matrix, including correct-rejection, as well as hits, misses, and false-positives (Table 3). Let *boundary-offset* be the absolute difference between the nearest boundary in detection result and every boundary in the ground-truth. The distribution over all testing trials is shown in Table 4.

Video	Argentina	KoreaA	KoreaB	Espana
Boundary	68	74	56	32
Non-Boundary	1369	1426	1468	869
*Hit	45.0	49.7	40.3	22.0
Miss	23.0	24.3	15.7	10.0
False Positive	19.0	23.0	21.7	14.0
Correct-Rejection	1350	1403	1446	855.0

Table 3. Boundary evaluations, **Hit* refers to boundaries detected within ± 3 seconds. Elements of the confusion matrix are the average testing result over 3 different training sets.

Offset (secs)	[0, 3]	(3, 6]	(6, 10]	(10,25]	(25,50]	>50
Percentage	62%	12%	5.8%	13%	6.7%	0.7%

Table 4. Boundary offset distribution

The results show that our classification scheme has consistent performance over various dataset; and models trained on one clip generalize well to other clips. The classification accuracy is above 80% for every clip, and more than 60% of the boundaries are detected within a 3-second ambiguity window(Table 4). Compared to the previously work [7], testing accuracy improves 1%, 15%, and 18% for clips KoreaB, Argentina and Espana (trained on KoreaA), respectively. Typical errors in the current algorithm are due to model breakdowns that feature values do not always reflect sematic state of the game, such as a brief switch of *play/break* without significant change in features. Moreover, if we regard *play* as the important content of a game, the algorithm suffers more from false alarms than misses, as we have observed that 91% of the *play*-segments are correctly classified on average.

5. CONCLUSION

In this paper, we presented new algorithms for soccer video segmentation and classification. First, *play* and *break* are defined as the basic semantic elements of a soccer video; second, observations of soccer video syntax are described and feature set is chosen based on these observations; and then, classification /segmentation is performed with HMM followed by dynamic programming. The results are evaluated in terms of classification accuracy and segmentation accuracy; extensive statistical analyses show that classification accuracy is about 83.5% over diverse data sets, and most of the boundaries are detected within a 3-second ambiguity window.

It is encouraging that high-level domain-dependent video structures can be computed with high accuracy using compressed-domain features and generic statistical tools. We believe that the performance can be attributed to the match of features to the domain syntax and the power of the statistical tools in capturing the temporal dynamics of the video.

The algorithms leaves much room for improvement and extension: (1) there are other relevant low-features that might provide complementary information and may help improve performance, such as camera motion, edge, audio, etc; (2) higher-level object detectors, such as goal and whistle detection, can be integrated; (3) further details of the content (e.g. different phases in a play) can be revealed by studying the structures within an HMM; (4) models that are more general and more capable for capturing interactions and temporal evolvement of features, objects, concepts and events can be used, such as dynamic Bayesian networks.

6. REFERENCES

[1] B. Burke and F. Shook, "Sports photography and reporting", Chapter 12, in *Television field production and reporting*, 2nd Ed, Longman Publisher USA, 1996

[2] Y. Gong; L.T. Sin; C. Chuan; H. Zhang; and M. Sakauchi, "Automatic parsing of TV soccer programs", *Proc. ICMCS'95*, Washington D.C, May, 1995

[3] J. Huang; Z. Liu; Y. Wang, "Joint video scene segmentation and classification based on hidden Markov model", *Proc. ICME 2000*, P 1551 -1554 vol.3, New York, NY, July 30-Aug3, 2000

[4] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE*, v 77 no 2, P 257 –286, Feb. 1989

[5] Rules of Soccer Game: http://www.scoresports.com/Tips/rules.htm

[6] V. Tovinkere , R. J. Qian, "Detecting Semantic Events in Soccer Games: Towards A Complete Solution", *Proc. ICME 2001*, Tokyo, Japan, Aug 22-25, 2001

[7] P. Xu, L. Xie, S.F. Chang, A. Divakaran, A, Vetro, and H. Sun, "Algorithms and system for segmentation and structure analysis in soccer video", *Proc. ICME 2001*, Tokyo, Japan, Aug 22-25, 2001

[8] D. Yow, B.L.Yeo, M. Yeung, and G. Liu, "Analysis and Presentation of Soccer Highlights from Digital Video" *Proc. ACCV, 1995*, Singapore, Dec. 5-8, 1995

[9] D. Zhong and S.F. Chang, "Structure Analysis of Sports Video Using Domain Models", *Proc. ICME 2001*, Tokyo, Japan, Aug 22-25, 2001