On Broadcasted Game Video Analysis: Event Detection, Highlight Detection, and Highlight Forecast

Wei-Ta Chu and Yung-Chieh Chou
National Chung Cheng University
Chiayi, Taiwan
wtchu@ccu.edu.tw, andy0983011@gmail.com

Abstract

Efficient access to broadcasted computer game videos is urgently demanded due to the emergence of live streaming platforms. The popularity of game video streaming builds a big market showing commercial potentials and arising many technical challenges. In this work we facilitate efficient access from two aspects: event detection and highlight detection. By recognizing designated text displayed on screen when important events occur, we associate game events with time stamps, and accordingly develop an interface to facilitate direct access. For highlight detection, we jointly consider visual features, event features, and viewer's behavior to construct two highlight models based on the psychophysiological approach and the data-driven approach, respectively. The concatenated highlights then enable compact game video presentation. To facilitate adaptive live streaming, a novel highlight forecast model is built to predict whether there will be a highlight in the next seconds, so that the streaming system can allocate more resource for more important segments on the fly. Comprehensive experiments based on various experimental settings demonstrate effectiveness of the proposed methods. We believe that this work is one of the early attempts on analyzing broadcasted computer game videos from the perspective of multimedia content analysis.

Index Terms: Broadcasted computer game video, event detection, highlight detection, highlight forecast, genetic algorithm, linear exponential smoothing

I. Introduction

Online live streaming platforms emerges rapidly in recent years. Streaming video services, including edited video programs like movies and TV shows, and live broadcast of events like sports, festival, and computer games, attract millions of users and cause significant network traffic on platforms like UStream[1], Livestream[2], and Twitch[3]. Among various types of videos, gaming videos, or the so-called electronic sports, is explosively growing in the aspects of the number of participants as well as the commercial market. It is also interesting that casual gamers were found to prefer watching professional gamers rather than playing the game by themselves [17]. In 2014 there were 100 millions unique users per month watching 16 billion minutes of streaming video on Twitch. In the same year over 11 millions videos were broadcasted on this platform per month. Such tremendous amounts of users and network traffic imply much commercial potential and many technical challenges. Among various issues, efficient access is obviously a key ingredient to making a streaming platform successful. In this work we will take *League of Legend* broadcasted on Twitch as an instance to investigate the efficient access issue, from the perspectives of events and highlights.

League of Legend (LoL) is a multiplayer online battle arena video game developed by Riot Games[4]. It is one of the most popular PC games in several continents such as Europe and North America. As of January 2014, there are 67 million gamers playing per month, 27 million gamers playing per day, and over 7.5 million gamers playing at the same time during each day's peak play time. Riot Games organizes the League of Legends Championship Series in many countries and continents. The biggest championship series attracted 32 million viewers online and granted the champion one million US dollars. From the aforementioned statistics, we see that this is a big business attracting tremendous amounts of users (gamers and viewers). From the viewpoint of multimedia research, explosive number of broadcasting videos and rich viewer behaviors, e.g., giving comments in the chat room, give rise to significant demands as well as research opportunities on efficient game video access, retrieval, and summarization.

In this work, we make an attempt to analyze game videos in order to facilitate efficient game video access and compact game representation. Three important components are described as follows.

- Event detection: Designated messages are shown on screen when important game events

[1]http://www.ustream.tv

[2]https://new.livestream.com/

[3]http://www.twitch.tv/

[4]http://www.riotgames.com

occur. We detect events through detecting and recognizing text displayed on screen, and then construct an index linking events and time stamps of game videos. Automatic text broadcast, therefore, can be achieved as the byproduct of event detection and can be used to facilitate efficient access.

- Highlight detection: Important events, prominent visual effects, as well as viewer's behavior are jointly considered to detect highlights. Two highlight detection methods are proposed: the psychophysiological approach based on the arousal model [15] and the data-driven approach based on support vector machine (SVM). Game highlight like the ones edited by professional reporters thus can be automatically generated to facilitate efficient browsing. In order to further improve highlight detection, we find the best subsegment among each candidate highlight and formulate it as an optimization problem solved by a genetic algorithm.
- Highlight forecasting: From the perspective of a streaming system, network bandwidth can be more efficiently utilized if streaming bitrates are dynamically adjusted depending on importance of video content. Based on feature characteristics and highlight modeling, we forecast whether there will be a highlight in the next few seconds so that the streaming server can accordingly adapt its streaming settings.

Event detection and highlight detection indeed have well-studied in many video domains. However, very few studies have been proposed for the newly-emerged game videos. It is worth mentioning that we propose a new application for a new type of video domain, especially considering the domain knowledge and special characteristics of game videos. One another novel component in highlight detection is the optimal subsegment selection based on a genetic algorithm (details to be described in Section IV-D). This component refines coarse results given by either the psychophysiological highlight detection approach or the data-driven highlight detection approach.

Highlight forecast, on the other hand, is relatively rarer in the literature. Although there were some works on *highlight prediction*, most of them focus on estimating the degree of highlight or determining whether a given video segment contains highlight or not. We, however, focus on *whether there will be a highlight in the next few seconds*. To our best knowledge, very few studies have been proposed on this issue.

Overall, the contributions of this work are threefold:

- We apply state-of-the-art event detection and highlight detection methods to a newly-emerged video domain and demonstrate practical applications.

- We formulate highlight refinement as an optimization problem and systematically solve it, in order to improve the results obtained by either the psychophysiological approach or the data-driven approach.
- We propose a novel highlight forecast application in order to facilitate adaptive game streaming, which would largely decrease cost of a streaming platform and simultaneously maintain user experience.

The rest of this paper is organized as follows. Sec. II provides literature survey. Sec. III describes details of event detection. Feature extraction and highlight models are described in Sec. IV, followed by the proposed highlight forecast model in Sec. V. Sec. VI provides performance evaluation from various perspectives, and Sec. VII concludes this paper.

II. Related Works

In this section we review related literature from three perspectives: live streaming systems, game video analysis, and video event detection and summarization.

A. Live Streaming Systems

As the emergence of live streaming systems, many works have been proposed to study such systems from various perspectives. Kaytoue et al. [17] focused on electronic sports videos streamed by Twitch and advocated that much potential revenue can be made to professional gamers, casters, and streaming platforms. They also showed that number of viewers is predictable and explainable. Pires and Simon [23] presented a dataset consisting of data collected from two main user-generated live streaming systems, i.e., Twitch and live service of YouTube. With this rich dataset, they studied overall bandwidth, number of unique channels, and popularity distribution in these systems. In [22], some observations from Twitch motivated them to implement adaptive bitrate streaming in order to reduce delivery bandwidth and to increase quality of experience of viewers. Hamilton et al. [14] presented an ethnographic investigation of the live streaming of video games on Twitch. They interviewed several Twitch users and found that difficulty of interaction influenced user's feeling. They explored the design problems and the implications of streaming systems as clues to improve not only the Twitch streaming system but also other streaming services.

B. Game Video Analysis

Studies designed for game videos, especially from the perspective of visual analysis, are quite few. Here we survey literature related to visual analysis for game videos. Douglass [9] utilized several image processing and computer vision techniques to show gameplay recording. For example, keyframes of game recording are shown in a grid manner, and many frames are superimposed to create average images showing recurrent visual artifacts. Lewis et al. [18] analyzed player's actions, such as actions per minute and spatial variance of action, to discover the correlation between actions and winning games. Not surprisingly, they found that gamers able to most quickly execute actions tend to win. Rioult et al. [25] extracted topological clues, such as the area of polygon where players move and the inertia of the team, to predict outcomes of multiplayer online battle arena games. Riegler et al. [24] developed a set of tools like zoom and drawing to annotate computer game videos, so that users can communicate general concepts of a specific game.

C. Video Event Detection and Summarization

Video event detection, highlight extraction, and summarization have been widely studied for years. In this section, we briefly describe some of the most recent works based on video genres, including sports videos, movies, TV shows, and consumer videos. Generally, no matter which video genre, the research trend starts from purely content-based analysis to adoption of external knowledge such as webcast text and social media. Most recently, crowdsourcing techniques and multiple resource integration also enable more advanced analysis.

The Bagadus system [26] seamlessly integrated data from multiple cameras mounted in a stadium and data from sensors on soccer players, and provided a real-time interaction subsystem for experts to annotate soccer events. This system enables a user to follow particular player(s), view events in the representation of panorama videos, and create video summaries. Annotating events by experts is expensive, and thus Sulser et al. [27] proposed to adopt the crowdsourcing technique to integrate annotations from crowd workers. A Bayesian network-based method was proposed to detect events for soccer videos in [29]. The proposed method captured dependencies among extracted features, based on the automatically learnt joint distribution of variables. In addition to conventional highlight events like goals and penalty kicks, Nguyen and Yoshitaka [21] proposed to include scenes of intensive competition and emotional moments in soccer video summaries. They measured interest level of a video clip based on cinematographic features and motion features. To annotate baseball videos, Chiu et al. [5] aligned high-level webcast text with video content in order to avoid unstable performance caused by purely content-based methods. For basketball videos, Hu et al. [16] proposed a robust player tracking system, and adopted player trajectories to detect highlight events and conduct tactic analysis. Chen and Chen [3] proposed a framework consisting of scoreboard detection, text/video alignment, and replay detection for basketball videos. Chen et al. [4] formulated video summarization as a discrete optimization problem and solved it by approaches originally proposed for resource allocation.

For movies, Evangelopoulos et al. [11] modeled time-varying perceptual importance of movies by fusing multimodal saliency, including auditory saliency derived from frequency analysis, visual saliency derived from intensity and color, and linguistic saliency derived from part-of-speech tagging. The multimodal saliency is then used to develop a generic video summarization algorithm. Tsai et al. [30] mined relationship between role-communities, and developed a movie summarization method based on social power of role-communities. Lu et al. [20] summarized movies from the auditory perspective. Important audio events such as cheer, laugh, and gunshot were detected and concatenated to form video summaries. Zhao et al. [34] summarized movies based on affective content analysis. Based on the extracted audio-visual affective features with a well-developed transformation, affective curves indicating types and intensities of emotions are formed, and the most affective video shots were selected in the summary.

Duan et al. [10] detected events in consumer videos by leveraging a large number of loosely labeled web images from multiple sources. They developed a decision function to select most relevant source domains and achieved much performance gain in event recognition. Dang and Radha [8] proposed an entropy-based measure of the heterogeneity of image patches, and utilized its temporal variation to achieve key frame extraction and video skimming for consumer videos. The idea of sparse coding reconstruction was adopted to do consumer video summarization in [7] and [33], while the former [7] used the entire video for reconstruction and needed much computation, and the latter [33] largely reduced computational cost by learning a dictionary by group sparse coding.

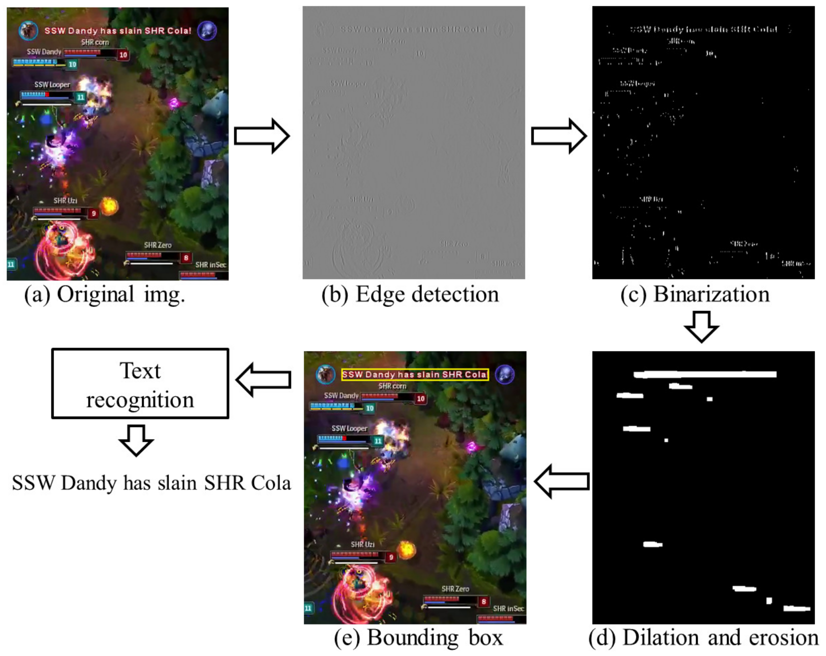
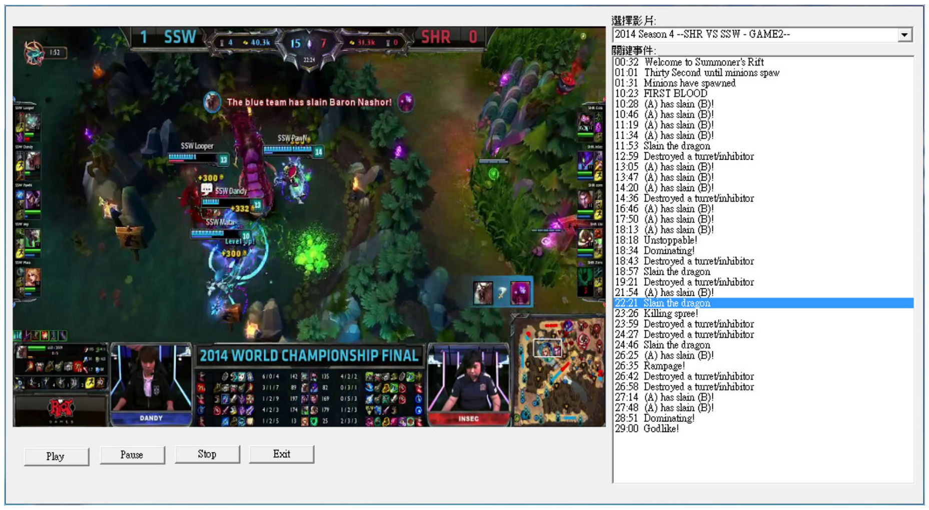


Fig. 1. The flowchart of event detection.

III. Event Detection

In League of Legends (LOL), important events like someone slaying others are presented as text messages on screen, as shown in Fig. 1(a). By recognizing this message and associating it with the corresponding time stamp, text broadcast showing the game progress can be generated. Fig. 2 shows the interface displaying the generated text broadcast of a game. Such presentation facilitates users to grasp the game progress at a glance and enables event-based access of the game video.

Fig. 1 shows the flowchart of event detection. For each video frame, we first apply the Sobel edge detector to extract edges (Fig. 1(b)), and conduct binarization to filter out weak edges (Fig. 1(c)). By morphological operations including dilation and erosion (Fig. 1(d)), more noisy edge pixels are filtered out, and the minimum bounding boxes of connected edge pixels are determined. Too small boxes are finally discarded (Fig. 1(e)). Although a more elegant bounding box determination method like the one proposed in [31] can be used, we found the proposed simple method already achieves satisfactory results.


1 SSW
SHR 0
The blue team has slain Baron Nashor!
2014 WORLD CHAMPIONSHIP FINAL
DANDY
INSEC
Play
Pause
Stop
Exit

選擇影片:
2014 Season 4 --SHR VS SSW - GAME2--
關鍵事件:

```
00:32  Welcome to Summoner's Rift
01:01  Thirty Second until minions spaw
01:31  Minions have spawned
10:23  FIRST BLOOD
10:28  (A) has slain (B)!
10:46  (A) has slain (B)!
11:19  (A) has slain (B)!
11:34  (A) has slain (B)!
11:53  Slain the dragon
12:59  Destroyed a turret/inhibitor
13:05  (A) has slain (B)!
13:47  (A) has slain (B)!
14:20  (A) has slain (B)!
14:36  Destroyed a turret/inhibitor
16:46  (A) has slain (B)!
17:50  (A) has slain (B)!
18:13  (A) has slain (B)!
18:18  Unstoppable!
18:34  Dominating!
18:43  Destroyed a turret/inhibitor
18:57  Slain the dragon
19:21  Destroyed a turret/inhibitor
21:54  (A) has slain (B)!
22:21  Slain the dragon
23:26  Killing spree!
23:59  Destroyed a turret/inhibitor
24:27  Destroyed a turret/inhibitor
24:46  Slain the dragon
26:25  (A) has slain (B)!
26:35  Rampage!
26:42  Destroyed a turret/inhibitor
26:58  Destroyed a turret/inhibitor
27:14  (A) has slain (B)!
27:48  (A) has slain (B)!
28:51  Dominating!
29:00  Godlike!
```

Fig. 2. The interface showing the generated text broadcast.

TABLE I
TEXT SHOWING IMPORTANT EVENTS IN LOL. THE TERMS (A) AND (B) COULD BE REPLACED WITH GAMER'S NAMES.

S_1	Welcome to Summoner's Rift!	S_2	Thirty seconds until minions spawn.
S_3	Minions have spawned.	S_4	First Blood!
S_5	(A) has slain (B)!	S_6	(A) is on a killing spree!
S_7	(A) is on a Rampage!	S_8	(A) is Unstoppable!
S_9	(A) is Dominating!	S_{10}	(A) is Godlike!
S_{11}	(A) is Legendary	S_{12}	The red team has slain the Dragon!
S_{13}	The red team has slain Baron Nashor!	S_{14}	(A) has destroyed a blue turret!
S_{15}	(A) has destroyed a blue inhibitor!	S_{16}	A minion has destroyed a blue turret!

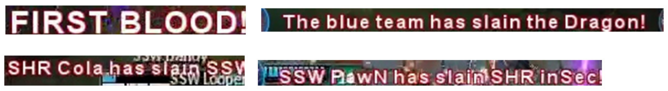

FIRST BLOOD!
The blue team has slain the Dragon!
SHR Cola has slain SSW
SSW PawN has slain SHR inSec!

Fig. 3. Samples of detected text regions. Top: regions with clear background; bottom: regions with cluttered background.

We employ the Tesseract OCR package[5] to recognize text in each detected bounding box. Let $W = \{w_1, ..., w_M\}$ be the set of recognized words. We compare W with predefined sentences S_1, ..., and S_{16} shown in Table I that would be displayed when the corresponding events occur. The bounding box consisting of W is claimed to represent the event i^* if $i^* = \arg\max_i |W \bigcap S_i|$, where $|\cdot|$ denotes the number of words in the input set. If recognized words in a box do not match with any of S_i, this box is viewed as noise and discarded.

The detected text regions are often with cluttered background, which impedes accurate text recognition. Fig. 3 shows samples of detected regions. The two regions at the top row are with clear background, while the two at the bottom row are with cluttered background. According to our experience, accuracy of the Tesseract OCR package on such data is only around 0.6. To resist to noises, we collect matching results in a duration and determine the occurred event by majority voting. The event text is usually displayed for 3 to 4 seconds. If a text region is recognized to contain S_i, we would check the recognition results of video frames in the following 4 seconds. This 4-second clip is determined to have the event j^* if $j^* = \arg\max_j \Delta_j$, where $\Delta_j = \sum_{k=1}^{K} \delta(f_k, S_j)$, $\delta(f_k, S_j) = 1$ if the frame f_k contains text S_j, and K is the number of video frames in the considered clip.

IV. Highlight Detection

To detect highlights in a game video, we extract features from both the video and viewers, and then construct highlight models based on the psychophysiological approach and the data-driven approach, respectively.

A. Features

Whether a video segment is attractive can be described from several perspectives. In this work, three types of features are extracted: visual features like motion intensity and frame dynamics, event features indicating the occurrence of events, and viewer's behavior features derived from chat logs. We divide a given game video into pieces of one-second segments, and extract the following features from each of them.

[5]https://code.google.com/p/tesseract-ocr/

- Motion intensity (G_1). Generally there is more motion when a highlight occurs, e.g., battle between two groups of gamers. We extract motion intensity between consecutive video frames, and average intensity values over all frames in a one-second segment. Higher motion usually indicates more interesting visual content. Suppose that the set of motion vectors between the ith frame and the $(i+1)$th frame is $U_i = \{\boldsymbol{u}_1, ..., \boldsymbol{u}_n\}$. The motion intensity between these two frames is defined as $\sum_{j=1}^{n} \|\boldsymbol{u}_j\|$, where $\|\boldsymbol{u}_j\|$ denotes the norm of the motion vector $\boldsymbol{u}_j$. More non-zero motion vectors with larger norms cause higher motion intensity.
- Frame dynamics (G_2). Special visual effects often occur when gamers invoke special attack abilities, and appearance of frames would suddenly change. We extract color histogram difference between frames, and average frame difference over all frames in a one-second segment. Usually larger frame dynamics indicates more interesting visual content.
- Number of gamers (G_3). Numbers of gamers involved in a battle is a strong clue to show how important this battle is. More gamers indicate more violent or more important events. Because name of each gamer is always displayed right above the corresponding avatar, as shown in Fig. 1(a), we can calculate the number of gamers appearing on screen by detecting text regions showing gamers' names. The region detection method mentioned in Sec. III is again adopted to detect text regions in appropriate sizes, i.e., not as large as the event text region, as well as not too small. We don't recognize words in these regions, because it is not necessary to know real names of these gamers. The number of detected bounding boxes is then viewed as the number of gamers appearing on screen. The average number of gamers over all frames of a one-second segment is then calculated as a feature.
- Event ratio (G_4). Occurrence of important events certainly indicates interesting content. Based on the events that have been detected by the method mentioned in Sec. III, we calculate the ratio of duration of events to the duration of a one-second segment as the feature. A larger ratio indicates that more events occur in this segment, and this segment is thus more attractive.
- Number of viewers chat (G_5). Considering user's behavior is important in event detection for social media platforms [13]. In addition to visual content, we also extract clues from chat logs given by viewers in online broadcasting. Chats directly reflect how viewers perceive the events just happened. More viewers chat or say praising words like "What a shot" or

"WOW" when important events occur. We thus calculate the number of speaking viewers per second as a feature. Note that the burst of chats emerge after important events occur. Therefore, the feature extracted from the tth segment actually indicates the importance of the $(t-b)$th segment. In the evaluation section, we will show this effect with varied b's.

- Number of emotion symbols (G_6). Twitch designs several emotion symbols to let viewers quickly express their feeling. We calculate the average number of emotion symbols over all frames of a one-second segment as the feature.

B. Arousal Model for Highlight Detection

We model game highlight by two approaches: the psychophysiological approach based on the arousal model [15] and the data-driven approach based on support vector machine (SVM). From psychophysiological experiments, when a user watches a video, the level of arousal rises as motion intensity increases. In this work, we investigate combining more clues in addition to motion to build the arousal model. As a data-driven approach, we can collect features extracted from highlighted/non-highlighted segments, and instead view highlight detection as a classification problem that can be solved by constructing an SVM classifier.

Before highlight detection, we detect shot change boundaries based on color histogram difference and edge change ratio [19]. We use one-second video segment as the unit for feature extraction and arousal model construction, while use video shot as the unit for SVM model construction.

Evolution of each feature mentioned above describes the level of arousal from one perspective. We can jointly consider all perspectives by appropriately combining them. Inspired by [15], the level of arousal of the kth video segment can be described as

$$A(k) = F(\hat{G}_i(k)), i = 1, ..., 6; k = 1, ..., N, \tag{1}$$

where $\hat{G}_i(k)$ is the ith feature value G_i after the smooth process, and the function F is for combining arousals from various perspectives. The smooth process is defined as

$$\hat{G}_i(k) = \frac{\max_k(G_i(k))}{\max_k(\tilde{G}_i(k))} \times \tilde{G}_i(k), \tag{2}$$

where $\tilde{G}_i(k)$ is the result of convolving the curve $G_i(k)$ with a Kaiser window of the length and shape parameter l and β, respectively, i.e., $\tilde{G}_i(k) = G_i(k) * K(l_i, \beta_i)$. The smooth process is designed to account for the degree of memory retention, and ensures that arousal does not change abruptly for consecutive video segments. The normalization defined in eqn. (2) makes different feature curves comparable, and eases curve integration. The function F in eqn. (1) can simply be a linear combination, and is defined as $F(\hat{G}_i(k)) = \frac{1}{6}\sum_{i=1}^{6}\hat{G}_i(k)$. Fig. 4 shows examples of arousal curves obtained from six features and the final integration curve.

Given a test video, we extract features from all one-second segments and construct arousal curves, and H highlight parts are detected by selecting H highest peaks of the integrated arousal curve. Suppose that the jth video segment v_j corresponds to the ith selected peak, the video segments preceding (v_p's) or following (v_q's) the jth video segment v_j are all selected as in the ith highlight H_i if they all belong to the same video shot $\mathcal{S}_j$:

$$\begin{aligned} H_i = \{ & v_p, v_j, v_q | v_p \in \mathcal{S}_j, v_q \in \mathcal{S}_j, \\ & p = j-1, j-2, ...; q = j+1, j+2, ...\}. \end{aligned} \tag{3}$$

C. SVM Model for Highlight Detection

In this model, we view highlight detection as a classification problem, i.e., classifying each test segment as a highlight or not. Here we use video shot as the analysis unit, and describe a shot by features extracted from one-second segments belonging to the same video shot. Mean, maximum, minimum, variance, and dynamic range (maximum minus minimum) of motion intensity in a shot, for example, are calculated. These derived features from G_1 to G_6 are concatenated as a 30-dimensional feature vector to represent a video shot.

We collect highlight/non-highlight video shots as positive/negative examples, and utilize the libSVM package [2] to train an SVM classifier with probability estimation. Given an unknown video shot, the SVM model determines whether it is a highlight shot or not. Note that we also can estimate the probability of being a highlight as well. This probability will be used in the highlight forecast model described later.

D. Optimal Subsegment Selection

The unit of highlights selected in the aforementioned methods is video shot, i.e., boundaries of a highlight certainly coincide with boundaries of shots. However, professional editors rarely limit themselves with shot boundaries when they select highlights. It is often that an edited highlight

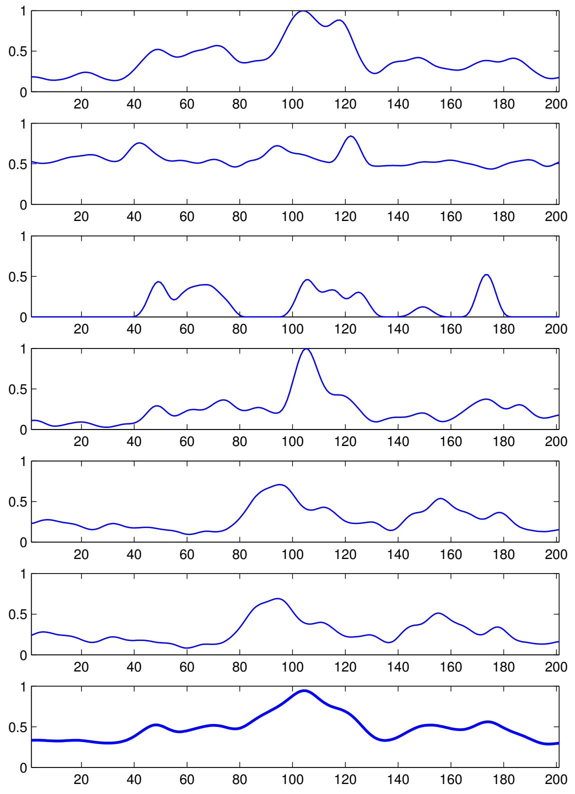


Fig. 4. Arousal curves constructed from different features (1st row to 6th row, $\hat{G}_1(k)$ to $\hat{G}_6(k)$), and the integrated curve (7th row, $A(k)$).

only includes the most important part of a video shot. Therefore, according to our preliminary experiments, we found that performance of highlight detection suffers from low precision value, i.e., the selected highlights are too long to be concise.

Given a "candidate" highlight selected by the arousal model or the SVM model, we would like to find a subsegment of it that most matches with the characteristics of true (manually edited) highlights. To narrow the gap between low-level features and high-level highlights, we adopt the idea of mid-level representation proposed in semantic concept analysis [6]. Features from G_1 to G_6 are first modeled by parametric models, from which parameters are viewed as mid-level features for building a high-level model. Details of the two-stage modeling approach are described below.

At the first stage, from the evaluation dataset we collect the true highlights $\{H_1, ..., H_N\}$ and extract associated features G_1 to G_6. We view all G_1 features, for example, extracted from the true highlights constitute a set, and then fit the distribution of G_1 by a Gaussian mixture model (GMM) $\mathcal{M}_1 = \sum_{i=1}^{M} w_i \mathcal{N}(\mu_i, \sigma_i)$, where w_i is the weight of the ith 1-dimensional Gaussian, and $\mathcal{N}(\mu_i, \sigma_i)$ is the ith Gaussian with mean μ_i and standard deviation σ_i, respectively. We view the parameters w_i, μ_i, and σ_i as mid-level features describing distribution of G_1. In this work, three Gaussian mixtures are used to build this GMM, i.e., $M = 3$, and thus in the G_1 aspect we totally extract $3 \times 3 = 9$ mid-level features (3 parameters for each mixture, and we have totally 3 mixtures). From the perspectives of G_1 to G_6, we describe all distributions by $9 \times 6 = 54$ mid-level features.

At the second stage, the collection of 54-dimensional mid-level feature vectors is again described by a mid-level Gaussian mixture model $\hat{\mathcal{M}}$, where the number of mixtures is also set as 3. We extract mid-level feature vectors from true highlights, and then adopt the expectation-maximization algorithm to construct the model $\hat{\mathcal{M}}$.

Given a candidate highlight starting from the (t_a)th second to the (t_b)th second, denoted as $[t_a, t_b]$, the problem of finding the optimal subsegment $[t_1^*, t_2^*]$ that best matches with the model $\hat{\mathcal{M}}$ is formulated as follows.

$$\begin{aligned} (t_1^*, t_2^*) = \arg\max_{t_1, t_2} \quad & p(\mathcal{F}([t_1, t_2])|\hat{\mathcal{M}}) \\ \text{subject to} \qquad & t_1 < t_2; t_a \leq t_1; t_2 \leq t_b; t_2 - t_1 > \epsilon \end{aligned} \tag{4}$$

where $\mathcal{F}(t_1, t_2)$ denotes the mid-level feature vector of the subsegment $[t_1, t_2]$. $p(\mathcal{F}([t_1, t_2])|\hat{\mathcal{M}})$ is the probability of the mid-level feature vector $\mathcal{F}([t_1, t_2])$ generated by the Gaussian mixture model $\hat{\mathcal{M}}$. The selected subsegment $[t_1^*, t_2^*]$ must start after t_a and ends before t_b, and its length must be no less then ϵ seconds. The optimal subsegment is found by maximizing the likelihood value. The parameter ϵ is dynamically set as the 80% of the total length of the candidate highlight. That is, if the length of the candidate highlight segment is T seconds, the parameter ϵ is $0.8 \times T$. This setting avoids selecting a very short subsegment from the candidate highlight.

We solve the aforementioned optimization problem by a *genetic algorithm*. If the length of the candidate highlight is T seconds, a potential subsegment $[t_1, t_2]$ is represented as a T-bit binary chromosome $\boldsymbol{c} = c_1, ..., c_T$, where $c_j = 0$ for $j = 1, ..., T$, except for $c_{t_1} = 1$ and $c_{t_2} = 1$. For example, 010010 denotes a subsegment starting at the 2nd second and ending at the 5th

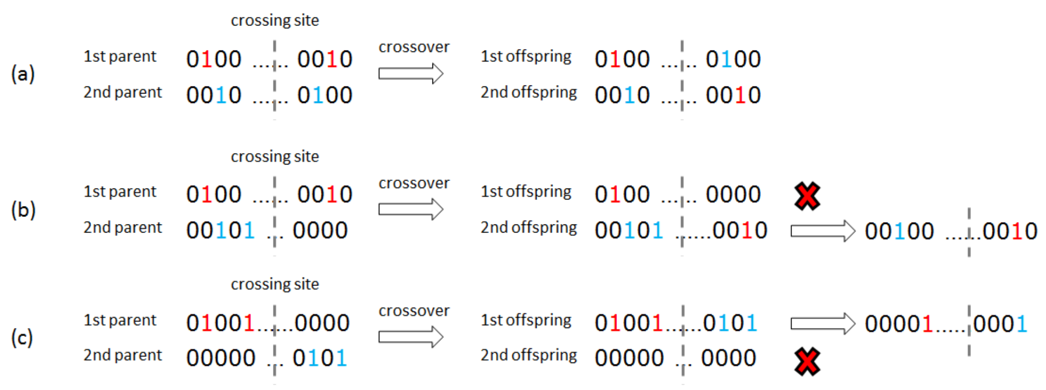

crossing site
(a)
1st parent 0100 0010
2nd parent 0010 0100
crossover
1st offspring 0100 0100
2nd offspring 0010 0010
(b)
1st parent 0100 0010
2nd parent 00101 .. 0000
1st offspring 0100 0000
2nd offspring 00101 0010
00100 0010
(c)
1st parent 01001.....0000
2nd parent 00000 .. 0101
1st offspring 01001.....0101
2nd offspring 00000 .. 0000
00001.....0001

Fig. 5. Illustration of the crossover operation and corresponding postprocessing.

second, from a six-second candidate highlight segment. We call the bit with 1 as a 1-bit in the following. To start the genetic algorithm, we initialize P chromosomes of length T to form the initial population, and evaluate the fitness (objective function value) of each chromosome by eqn. (4). We then select chromosomes and apply *crossover* and *mutation* operations to generate the next-generation population. The processes of fitness evaluation, chromosome selection, and next-generation population generation keeps iterating until the predefined stop criterion meets. In this work, we run the process 100 times and select the optimal subsegment corresponding to the chromosome with the maximum fitness.

About chromosome selection, we form a mating pool by selecting chromosome one by one from the initial population based on the roulette-wheel selection scheme. That is, a chromosome $\boldsymbol{c}_k$ is selected into the mating pool according to the probability $f(\boldsymbol{c}_k)/\sum_{k=1}^{P} f(\boldsymbol{c}_k)$, where the numerator $f(\boldsymbol{c}_k)$ is the fitness of the chromosome $\boldsymbol{c}_k$, and the denominator is the total fitness of the initial population. This scheme more likely selects *better* (with higher fitness) chromosomes to generate the next population. In this work, totally 60 chromosomes, i.e., $P = 60$, are selected into the mating pool. More or less numbers of chromosomes can be selected, and the genetic algorithm can still get convergence with appropriate number of iteration.

We randomly select a pair of chromosomes from the mating pool as the parents, and employ the *crossover* operation to generate a pair of offspring chromosomes. Let us denote the two chromosome as $\boldsymbol{c} = c_1, ..., c_T$ and $\boldsymbol{c}' = c'_1, ..., c'_T$. A number between 2 and $T - 1$ is randomly chosen, say s, to be the location of the *crossing site*. We then perform the crossover operation by exchanging substrings of the parents to the left of the crossing site. That is, one offspring chromosome is $\hat{\boldsymbol{c}} = c_1, ..., c_{s-1}, c'_s, c'_{s+1}, ..., c'_T$, and another is $\hat{\boldsymbol{c}}' = c'_1, ..., c'_{s-1}, c_s, c_{s+1}, ..., c_T$. The crossover operation conceptually exchanges the ways two chromosomes define the beginnings and the endings of selected subsegments.

Note that the generated offspring chromosomes not necessarily contain exactly two 1-bits. There are four cases after the crossover operation. (1) If there is exactly one 1-bit on the left side and on the right side of the crossing site, no future postprocessing is needed, as shown in the first case of Fig. 5. (2) If any one side of the crossing site contains no 1-bit, this offspring chromosome is said to be invalid and is discarded, as shown in the first offspring of Fig. 5(b) and the second offspring of Fig. 5(c). (3) The left side of the crossing site contains two 1-bits and the right side contains only one 1-bit. The two 1-bits at the left side turn out to come from the same partent, and we randomly select one of them and set it as 0 to ensure only one 1-bit appears at each side. The second offspring chromosome of Fig. 5(b) shows this case. (4) If both sides of the crossing site contain two 1-bits, for each side we randomly select one of them and set it as 0. The first offspring chromosome of Fig. 5(c) shows this case.

After the crossover operation, we apply the *mutation* operation by randomly moving the location of 1-bit with a given (small) probability. Taking a chromosome $\boldsymbol{c} = c_1, ..., c_T$, where $c_{t_1} = 1$ and $c_{t_2} = 1$, $1 < t_1 < t_2 < T$ as an instance, if the 1-bit at t_1 is decided to be mutated, it is set as 0, and one another bit located between 1 and T, except for that at t_1 and t_2, is randomly selected and set as 1.

V. Highlight Forecast

Although live game streaming is quite popular in industry, there is still much room to improve user experience. Zhang and Liu [32] reported that Twitch viewers suffer from ≥ 12 seconds lags on average, which is unacceptable because real-time interactivity is required between streamers and viewers. To improve user experience, Twitch was forced to build new date centers and upgrade existing ones. In [22], Twitch was estimated to consume 1Tbps bandwidth on average in 2014, and that cost Twitch over 10 million dollars per month. Therefore, how to save bandwith without sacrificing user experience becomes an important issue.

To tackle this problem, one of the best strategies for a streaming platform is to dynamically adjust streaming bitrates so that the delivery bandwidth can be reduced and quality of experience of viewers can be maintained. Recently, Dynamic Adaptive Streaming over HTTP (DASH) was implemented on Twitch to verify the effectiveness of adaptive streaming [22]. They proposed two simple strategies respectively based on number of viewers and popularity of channel to adapt streaming bitrate. We advocate that adapting bitrate based on game content is another interesting alternative. Fan-Chiang et al. [12] proposed a segment-of-interest driven live game streaming system, which dynamically allocates more resources to more important segments. This system coincides with our concept: if we can forecast whether there will be a highlight in a few seconds, this information could be the key for the server to allocate resources.

To forecast a highlight, a simple method is that we assume the probability of highlight occurrence is the same in a locality. The local mean probability at the tth second is calculated as

$$m_t = \alpha h_t + (1-\alpha) m_{t-1}, \tag{5}$$

where h_t is the probability of highlight occurrence at the tth second, given by the SVM-based highlight detection model. The parameter α is a smooth constant controlling the balance between the current observation h_t and the previous smooth value m_{t-1}. The probability of highlight occurrence at the $(t+1)$th second can be simply estimated as $\hat{h}_{t+1} = m_t$.

The aforementioned method assumes there is no trend in the data. However, more visual clues, such as gamers gathering together, appear when a highlight tends to occur. Therefore, we adopt Brown's linear exponential smoothing method [1] that can more effectively capture the time-varying trend to forecast highlight probability. Let m' denote the singly-smoothed series obtained by applying the simple smooth process mentioned above. That is,

$$m'_t = \alpha h_t + (1-\alpha) m'_{t-1}, \tag{6}$$

where h_t is the probability of highlight occurrence at the tth second, given by the SVM-based highlight detection model. Let m'' denote the doubly-smoothed series obtained by applying the simple smooth process to series m':

$$m''_t = \alpha m'_t + (1-\alpha) m''_{t-1}. \tag{7}$$

The probability of highlight occurrence at the $(t+k)$th second is estimated as

$$\hat{h}_{t+q} = L_t + kT_t, \tag{8}$$

where $L_t = 2m'_t - m''_{t-1}$ is the estimated level at the tth second, and $T_t = (\alpha/(1-\alpha))(m'_t - m''_{t-1})$ is the estimated trend at the tth second. This estimation model starts with $m'_1 = m''_1 = h_1$, i.e., setting both series equal to the first observed value at $t = 1$. By jointly considering the estimated level and the trend, we estimate the probability of highlight that will occur q seconds later. In the evaluation section, we will evaluate accuracy of highlight forecast with different settings on q.

Note that we need to carefully examine the probability h_t. Because solely estimating this probability based on features in a one-second segment is too limited and often incurs worse performance, we actually estimate h_t based on features extracted from the $(t-p)$th second to the tth second. The influence of the parameter p on highlight forecast performance will be shown in the evaluation section.

VI. Evaluation

A. Evaluation Dataset

We collect 24 games of 2014 League of Legends World Champion that were broadcasted by Twitch as the evaluation dataset. As the highlight ground truth, we collect manually-edited highlights of each game that were produced by a team[6] constituted by professional gamers. Table II shows average lengths of six game series and the edited highlights. To evaluate performance of text broadcast generation, we also collect text broadcast corresponding to each game from a computer game website[7]. Availability of these data in different web platforms shows that these games widely attracted gamers around the global. To evaluate performance of event detection, we examine each game video and manually label events and their corresponding timestamps.

B. Automatic Text Broadcast

Performance evaluation of text broadcast can be conducted from two perspectives: overlap between the detected events and manual text broadcast, and accuracy of the detected events. From the first perspective, we manually examine the overlap between text broadcast ground truth and detected events. It is worth emphasizing that the number of detected events is often

[6]https://www.youtube.com/user/Kazawuna/about

[7]http://www.tgbus.com/

TABLE II

STATISTICS OF THE EVALUATION DATASET.

Videos	Avg. Length (mm:ss)	Avg. Length of Highlight (sec.)
2014 NWS[a] VS OMG GAME 1–3	51:28	550
2014 SHR VS EDG GAMES 1–5	40:27	459
2014 SHR VS OMG GAMES 1–5	44:08	594
2014 SHR VS SSW GAMES 1–4	34:36	441
2014 SSB VS C9 GAMES 1–4	42:28	628
2014 SSB VS SSW GAMES 1–3	35:42	517
Average	40:35	531

[a]NWS, OMG, SHR, EDG, SSW, SSB, and C9 are all team names.

more than that mentioned in the manually edited text broadcast. The text broadcaster often summarizes several consecutive events into a single battle due to space limitation. Therefore, we define the *hit rate* of detected events as the ratio of the number of detected events that are really included in records of text broadcast. Table III shows average hit rates of detected events in different game series. As can be seen from this table, over 90% of detected events are really included in manual text broadcast, which serve as good clues for highlight detection.

From the second perspective, we calculate the average precision values (the ratio of the number of detected events that are really important events to that of all detected events) and the average recall values (the ratio of the number of detected events that are really important events to that of all events defined in ground truths) of detected events in each game series (Table IV). From this table we clearly see that the proposed event detection method achieves very high precision. Relatively worse performance is obtained in terms of recall, due to mis-detection caused by extremely cluttered background. Moreover, sometimes several consecutive events occur intensively with dazzling visual artifacts, and our method fails to detect all of them.

Fig. 6 shows visual samples of three detected events. From top to bottom, these samples are: A gamer is on a rampage (S_7 shown in Table I); a gamer is unstoppable! (S_8); and a gamer has slain another gamer (S_5). We can see that event messages are shown on the screen, very probably with cluttered background.

TABLE III

AVERAGE HIT RATES OF DETECTED EVENTS, CALCULATING BASED ON OVERLAP BETWEEN THE TEXT BROADCAST GROUND TRUTH AND DETECTED EVENTS.

Videos	Average Hit Rates
2014 NWS VS OMG GAME 1–3	0.89
2014 SHR VS EDG GAME 1–5	0.93
2014 SHR VS OMG GAME 1–5	0.92
2014 SHR VS SSW GAME 1–4	0.89
2014 SSB VS C9 GAME 1–4	0.96
2014 SSB VS SSW GAME 1–3	0.95
Overall	0.92

TABLE IV

AVERAGE PRECISION AND RECALL VALUES OF DETECTED EVENTS.

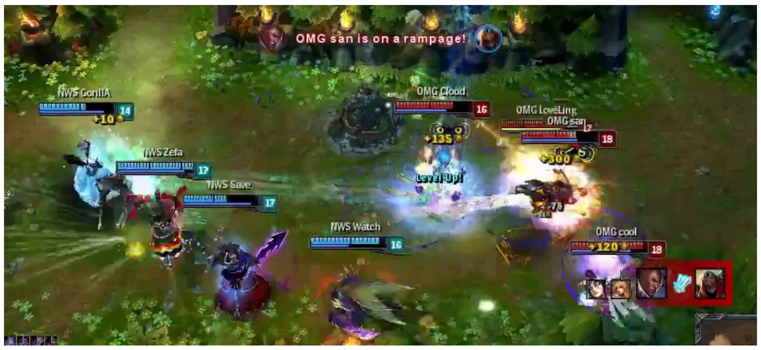
Videos	Average Precision	Average Recall
2014 NWS VS OMG GAME 1–3	0.98	0.87
2014 SHR VS EDG GAME 1–5	0.99	0.78
2014 SHR VS OMG GAME 1–5	1.00	0.82
2014 SHR VS SSW GAME 1–4	0.98	0.74
2014 SSB VS C9 GAME 1–4	0.98	0.79
2014 SSB VS SSW GAME 1–3	0.99	0.78
Overall	0.99	0.79

C. Highlight Detection

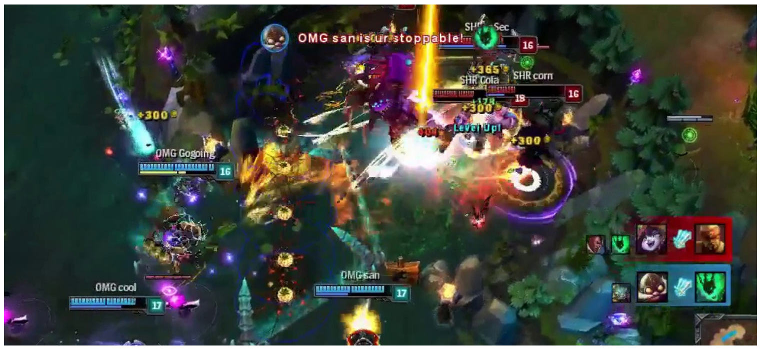
Performance measurement. Performance of highlight detection is measured in terms of precision, recall, and F-measure. Let $\mathcal{G}$ denote the set of true highlight segments and $\mathcal{D}$ the set of detected highlight segments. The precision rate is calculated as $p(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_j|}$, where $C_i \in \mathcal{G}$ and $C_j \in \mathcal{D}$. The notation $|\cdot|$ denotes the length in term of seconds. The recall rate is calculated as $r(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i|}$. By jointly considering precision and recall, the F-measure F is calculated as:

$$F = \frac{1}{Z} \sum_{C_i \in \mathcal{G}} |C_i| \max_{C_j \in \mathcal{D}} \{ f(C_i, C_j) \}, \tag{9}$$

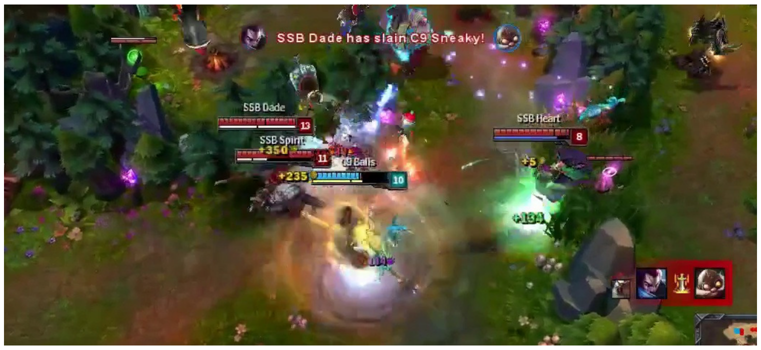
$$f(C_i, C_j) = \frac{2 \times p(C_i, C_j) \times r(C_i, C_j)}{p(C_i, C_j) + r(C_i, C_j)}, \tag{10}$$



(a)



(b)



(c)

Fig. 6. Visual samples of three detected events. From top to bottom: A gamer is on a rampage (S_7 shown in Table I); a gamer is unstoppable! (S_8); and a gamer has slain another gamer (S_5).

The value $Z = \sum_{C_i \in \mathcal{G}} |C_i|$ is the normalization factor. Higher F-measure means better detection performance.

Feature Settings. In the following, we first evaluate the influence of different feature settings on highlight detection based on the arousal model. As mentioned in Sec. IV-A, chats usually largely emerge after some events happen for a while. We evaluate F-measure of detection results

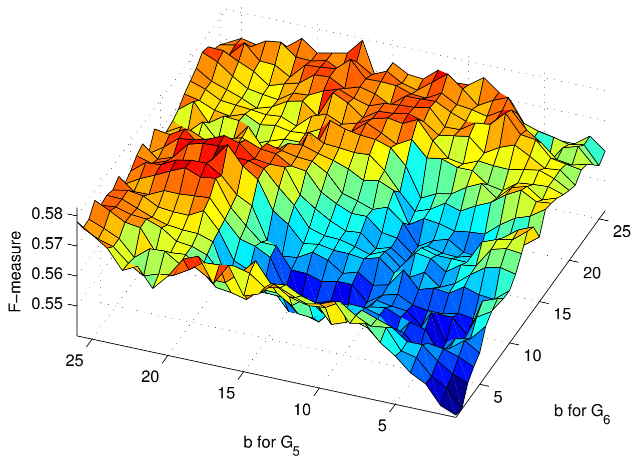


Fig. 7. Backtracking parameters vs. precision of highlight detection.

based on different backtrack settings, i.e., the parameter b mentioned in the description of G_5 and G_6 (Sec. IV-A), and show performance variations in Fig. 7. From this figure, the best detection performance can be achieved if we backtrack 20 seconds for G_5 and 11 seconds for G_6, respectively. That means, G_5 and G_6 of the tth video segment (at the tth second) is actually extracted from the text at the $(t + 20)th$ second and at the $(t + 11)th$ second, respectively. The difference between settings for G_5 and G_6 is not surprising, because viewers usually type predefined emotion symbols just after some events happen, and then type text comment. We use these backtrack settings in the following experiments.

Generally three types of features are used in highlight detection, i.e., visual features (G_1 to G_3), event feature (G_4), and viewer behavior features (G_5 and G_6). We separately evaluate each type of features as well as jointly consider all of them, and show highlight detection performance in Table V. Comparing the first three rows shows that visual features are the most robust, which is not surprising because viewer's behavior (chat) is unlimited and is often noisy. By jointly considering all features, the best highlight detection performance can be obtained.

Smooth Settings. The influence of the size of Kaiser window, which simulates human's short-term memory and reduces noise, on highlight detection performance is shown in Fig. 8. Overall, detection performance sharply increases as the window size increases from 5 seconds to 15 seconds, and is gradually decreasing as the window size increases further. From this figure we

TABLE V

PERFORMANCE OF HIGHLIGHT DETECTION BASED ON THE AROUSAL MODEL WITH DIFFERENT FEATURE SETTINGS.

Features	Precision	Recall	F-measure
G_1, G_2, G_3 (visual)	0.501	0.584	0.540
G_4 (event)	0.495	0.540	0.517
G_5 and G_6 (chat)	0.437	0.538	0.482
G_1 to G_6	**0.545**	**0.626**	**0.583**

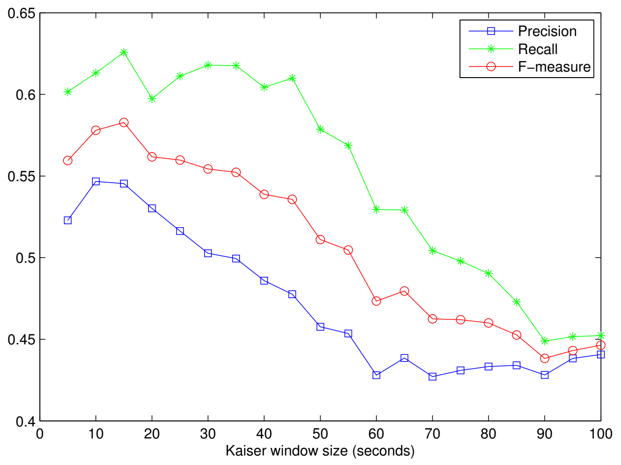


Fig. 8. Kaiser window size vs. performance of highlight detection.

set the size of Kaiser window for constructing arousal curves as 15 seconds in the following experiments.

Peak Selection and Segment Boundaries. After the integrated arousal curve is constructed, appropriate number of peaks are selected and the corresponding video segments are extracted as game highlights. Therefore, two problem arises: how many peaks should we pick, and how to determine the video segment corresponding to each peak? We design two schemes to deal with these problems.

- Scheme 1: We select the 16 highest peaks. For a peak located at the tth second, the video segment ranging from $t-17$ seconds to $t+17$ seconds is extracted as the highlight. Sixteen extracted video segments corresponding to the 16 selected peaks are concatenated as the final highlight. The number of selected peaks and the setting of selected ranges are from statistics of the data collection, i.e., there are averagely 16 highlight segments in a game,

TABLE VI

PERFORMANCE OF HIGHLIGHT DETECTION BASED ON THE AROUSAL MODEL WITH DIFFERENT PEAK SELECTION SCHEMES.

Features	Precision	Recall	F-1 measure
Scheme 1	0.486	0.653	0.557
Scheme 2	0.545	0.626	0.583

and each segment averagely lasts 34 seconds.

- Scheme 2: An iterative selection scheme is proposed here. Let $\{P_1, ..., P_N\}$ denote the set of N peaks that have already been selected. The average peak value is calculated as $\bar{P} = \frac{1}{N}\sum_{i=1}^{N} P_i$. The $(N+1)$th highest peak P_{N+1} will also be selected as highlight if $P_{N+1} > 0.8 \times \bar{P}$. We sequentially select highest peaks until the next highest peak does not meet the criterion. For a peak located at the tth second, we select the video shot containing the tth second as the highlight. This selection design makes boundaries of highlight segments coincide with shot boundaries.

Table VI shows highlight detection performance based on the two different peak selection schemes. As can be seen, although Scheme 1 gives highlights that best match with the average statistics from the training data, the adaptive designs in Scheme 2 yields much better precision and thus yields better detection performance in terms of F-measure.

Arousal model vs. SVM model. After the investigation mentioned above, we adopt the best feature settings to extract features, and accordingly train the SVM model for highlight detection. In the following experiment, we adopt the five-fold cross validation scheme to evaluate the SVM approach. Table VII shows highlight detection performance comparison between the arousal model and the SVM model. The rows with SS in the parentheses denote obtained performance with subsegment selection mentioned in Sec. IV-D. From this table we can see that, although both models yields similar precision rates, the SVM model achieves much higher recall rates. In the arousal model, we smooth feature values to construct arousal curves. The smooth operation reduces noises but also causes information loss, and might be the reason of lower recall rate. Based on this experiment, we conclude that the SVM model generally outperforms the arousal model. Comparing performance with and without subsegment selection, we observe that more performance gain can be achieved for the SVM model. Another observation is that the

TABLE VII

PERFORMANCE OF HIGHLIGHT DETECTION BASED ON THE AROUSAL MODEL AND THE SVM MODEL.

Models	Precision	Recall	F-1 measure
Arousal model	0.545	0.626	0.583
Arousal model (SS)	0.557	0.620	0.587
SVM model	0.520	0.821	0.637
SVM model (SS)	0.535	0.806	0.644

subsegment selection process only gives marginal improvement over the baseline cases. After careful examination, we found that the main problem is not the malfunction of the optimization algorithm, but the effectiveness of features. Integrating more effective game-specific features is thus an important research direction for the future.

Tolerance on detection performance. In eqn. (9) and eqn. (10), precision, recall, and F-measure can be unity only when the detected highlight is 100% overlapped with the ground truth. However, professional reports usually cut video shots at the time instant right before/after important events occur. On the other hand, what viewers want is compact representation, and viewers are usually satisfied with detected highlights with more than 50% overlapping with the “real highlights”. Motivated by the PASCAL VOC object detection task, where a correct detection is claimed if the detected object is overlapped with the ground truth by over 50%, we evaluate detection performance variations with various tolerance settings. The tolerance factor is 0 in eqn. (9) and eqn. (10), and is 0.3 if we view a detected highlight with more than 70% overlapping with the ground truth as a correct detection.

Fig. 9 shows performance variations based on different tolerance factors. As expected, performance increases if we allow more tolerance. For the arousal model and the SVM model, the F-measures respectively increase to 0.70 and 0.77 if we set the tolerance factor as 0.5. These results are promising and show the effectiveness of both highlight models.

Fig. 10 shows selected keyframes from a highlight automatically detected from 2014 SHR vs. OMG Game 3. This was a fierce battle starting from the gamer “Cloud” of team OMG slew the gamer “Uzi” of team SHR (Fig. 10(b)), followed by the gamer “Loveling” killing “Zero” (Fig. 10(c)). The team SHR then struck back by the gamer “corn” consecutively killing two rivals (Fig. 10(e) and Fig. 10(f)). The gamer “Gogoing” of team OMG then aggressively

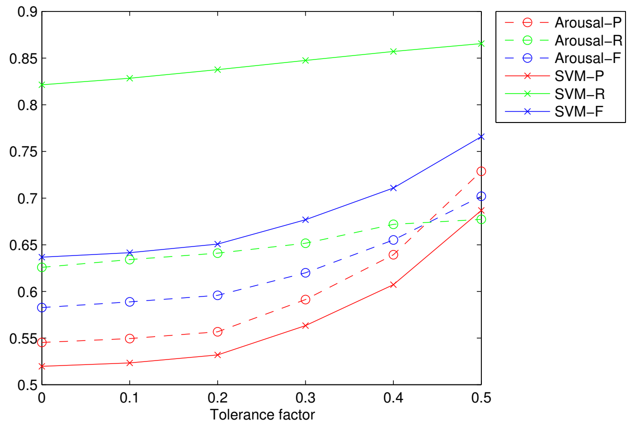
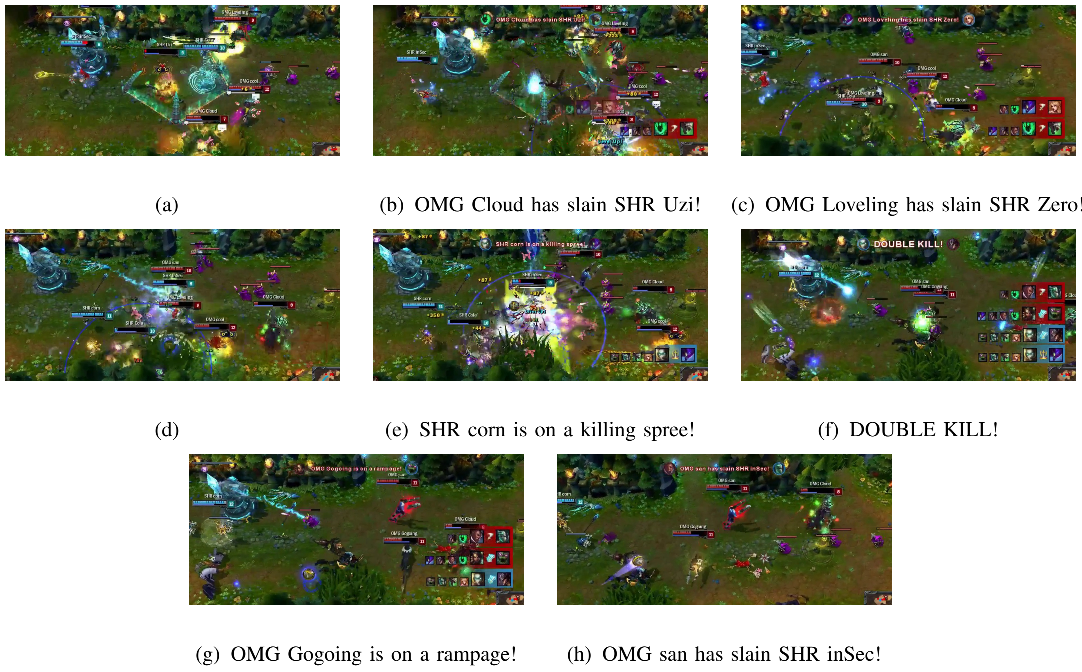


Fig. 9. Detection performance vs. different tolerance settings for highlight detection.



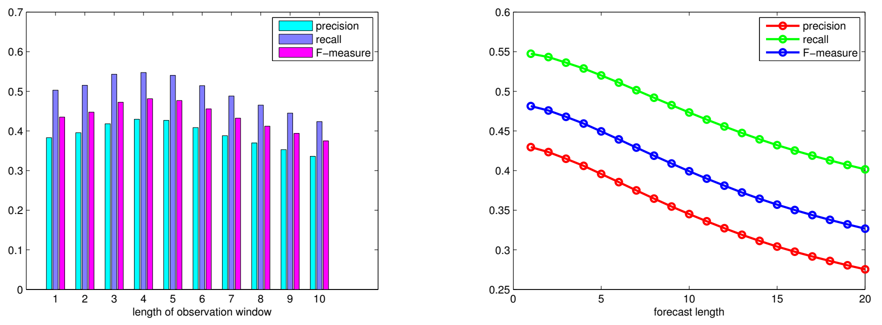
(a) (b) OMG Cloud has slain SHR Uzi! (c) OMG Loveling has slain SHR Zero!

(d) (e) SHR corn is on a killing spree! (f) DOUBLE KILL!

(g) OMG Gogoing is on a rampage! (h) OMG san has slain SHR inSec!

Fig. 10. Sample keyframes of a detected highlight.

sprinted (Fig. 10(g)), and the whole battle ends with the gamer “san” killing the rival “inSec” (Fig. 10(h)). Overall, the team SHR collapsed after this big battle.



(a) (b)

Fig. 11. Performance of highlight forecast. (a) Performance variation based on different settings of observations. (b) Performance variation based on different settings of forecast.

D. Highlight Forecast

We evaluate highlight forecast from the classification viewpoint. We forecast that there will be a highlight occurring at the $(t+q)$th second if the estimated probability $\hat{h}_{t+q}$ is larger than 0.5. Therefore, each one-second segment is labeled with highlight or non-highlight, and is compared with the ground truth. Precision, recall, and F-measure thus can be calculated to show forecast performance, and are shown in Fig. 11. Fig. 11(a) shows forecast performance with different settings of observations, i.e., the length of observation window p (mentioned in Sec. V) where features are extracted. Overall, forecast based on features extracted from the $(t-4)$th second to the tth second yields the most accurate results. The following experiment thus uses the four-second-segment features to implement forecast. Fig. 11(b) shows performance variation based on different settings of forecast, i.e., the forecast length q mentioned in Sec. V. As expected, forecast performance decreases as the forecast length increases.

VII. Conclusion

We have presented three components to facilitate efficient access to broadcasted computer game videos. Through recognizing designated text displayed on screen when events occur, we detect game events and build an interface to ease direct access. For game highlight detection, we propose features to describe visual appearance, events, and viewer's behavior, and then construct two highlight models based on the psychophysiological approach and the data-driven approach, respectively. A highlight forecast method is further proposed to estimate whether there will be a highlight in a few seconds, to facilitate dynamic game video streaming. Evaluation on famous game videos shows that the proposed methods yield accurate event detection and promising highlight detection performance. In the future, performances of event detection or highlight detection are to be boosted by considering more elegant features. In addition, adopting a learning model that automatically learns features and parameters to describe highlights will be another research direction [28].

REFERENCES

[1] R. G. Brown. Exponential smoothing for predicting demand. *Cambridge, Massachusetts: Arthur D. Little Inc.*, 1956.

[2] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(27):1–27, 2011.

[3] C.-M. Chen and L.-H. Chen. Novel framework for sports video analysis: A basketball case study. In *Proceedings of IEEE International Conference on Image Processing*, pages 961–965, 2014.

[4] F. Chen, C. De Vleeschouwer, and A. Cavallaro. Resource allocation for personalized video summarization. *IEEE Transactions on Multimedia*, 16(2):455–469, 2014.

[5] C.-Y. Chiu, P.-C. Lin, S.-Y. Li, T.-H. Tsai, and Y.-L. Tsai. Tagging webcast text in baseball videos by video segmentation and text alignment. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(7):999–1013, 2012.

[6] W.-T. Chu, W.-H. Cheng, J. Y.-J. Hsu, and J.-L. Wu. Towards semantic indexing and retrieval using hierarchical audio models. *Multimedia Systems*, 10(6):570–583, 2005.

[7] Y. Cong, J. Yuan, and J. Luo. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Transactions on Multimedia*, 14(1):66–75, 2012.

[8] C. T. Dang and H. Radha. Heterogeneity image patch index and its application to consumer video summarization. *IEEE Transactions on Image Processing*, 23(6):2704–2718, 2014.

[9] J. Douglass. Computer visions of computer games: Analysis and visualization of play recordings. In *Proceedings of Workshop on Media Arts, Science, and Technology: The Future of Interactive Media*, 2009.

[10] L. Duan, D. Xu, and S.-F. Chang. Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1338–1345, 2012.

[11] G. Evangelopoulos, A. Zlatintsi, A. Potamianos, P. Maragos, K. Rapantzikos, G. Skoumas, and Y. Avrithis. Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention. *IEEE Transactions on Multimedia*, 15(7):1553–1568, 2013.

[12] T.-Y. Fan-Chiang, H.-J. Hong, and C.-H. Hsu. Segment-of-interest driven live game streaming: Saving bandwidth without degrading experience. In *Proceedings of International Workshop on Network and Systems Support for Games*, pages 1–6, 2015.

[13] Y. Gao, S. Zhao, Y. Yang, and T.-S. Chua. Multimedia social event detection in microblog. In *Proceedings of International Conference on Multimedia Modeling*, pages 269–281, 2015.

[14] W. A. Hamilton, O. Garretson, and A. Kerne. Streaming on twitch: Fostering participatory communities of play within live mixed media. In *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 1315–1324, 2014.

[15] A. Hanjalic and L.-Q. Xu. Affective video content representation and modeling. *IEEE Transactions on Multimedia*, 7(1):143–154, 2005.

[16] M.-C. Hu, M.-H. Chang, J.-L. Wu, and L. Chi. Robust camera calibration and player tracking in broadcast basketball video. *IEEE Transactions on Multimedia*, 13(2):266–279, 2011.

[17] M. Kaytoue, A. Silva, L. Cerf, W. Meira Jr., and C. Raissi. Watch me playing, i am a professional: a first study on video game live streaming. In *Proceedings of International Conference Companion on World Wide Web*, pages 1181–1188, 2012.

[18] J. M. Lewis, P. Trinh, and D. Kirsh. A corpus analysis of strategy video game play in starcraft: Broodwar. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, pages 687–692, 2011.

[19] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Proceedings of SPIE Storage and Retrieval for Still Image and Video Databases VII*, volume 3656, pages 290–301, 1999.

[20] T. Lu, Y. Weng, and G. Wang. Audiotory movie summarization by detecting scene changes and sound events. In *Proceedings of International Conference on Pattern Recognition*, pages 756–760, 2014.

[21] N. Nguyen and A. Yoshitaka. Soccer video summarization based on cinematography and motion analysis. In *Proceedings of IEEE International Workshop on Multimedia Signal Processing*, 2014.

[22] K. Pires and G. Simon. Dash in twitch: Adaptive bitrate streaming in live game streaming platforms. In *Proceedings of Workshop on Design, Quality and Deployment of Adaptive Video Streaming*, pages 13–18, 2014.

[23] K. Pires and G. Simon. Youtube live and twitch: A tour of user-generated live streaming systems. In *Proceedings of ACM Multimedia Systems Conference*, pages 225–230, 2015.

[24] M. Riegler, M. Lux, V. Charvillat, A. Carlier, R. Vliegendhart, and M. Larson. Videojot: A multifunctional video annotation tool. In *Proceedings of ACM International Conference on Multimedia Retrieval*, pages 534–537, 2014.

[25] F. Rioult, J.-P. Metivier, B. Helleu, N. Scelles, and C. Durand. Mining tracks of competitive video games. *AASRI Procedia*, 8:82–87, 2014.

[26] H. K. Stensland, V. R. Gaddam, M. Tennoe, E. Helgedagsrud, M. Naess, H. K. Alstad, A. Mortensen, R. Langseth, S. Ljodal, O. Landsverk, C. Griwodz, P. Halvorsen, M. Stenhaug, and D. Johansen. Bagadus: An integrated real-time system for soccer analytics. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 10(1s):Article No. 14, 2014.

[27] F. Sulser, I. Giangreco, and H. Schuldt. Crowd-based semantic event detection and video annotation for sports videos. In *Proceedings of ACM Workshop on Crowdsourcing for Multimedia*, pages 63–68, 2014.

[28] M. Sun, A. Farhadi, and S. Seitz. Ranking domain-specific highlights by analyzing edited videos. In *Proceedings of Eurpean Conference on Computer Vision*, pages 787–802, 2014.

[29] M. Tavassolipour, M. Karimian, and S. Kasaei. Event detection and summarization in soccer videos using bayesian network and copula. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(2):291–304, 2014.

[30] C.-M. Tsai, L.-W. Kang, C.-W. Lin, and W. Lin. Scene-based movie summarization via role-community networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(11):1927–1940, 2013.

[31] M. Wang, Y. Sheng, B. Liu, and X.-S. Hua. In-image accessibility indication. *IEEE Transactions on Multimedia*, 12(4):330–336, 2010.

[32] C. Zhang and J. Liu. On crowdsourced interactive live streaming: A twitch.tv-based measurement study. In *Proceedings of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 55–60, 2015.

[33] B. Zhao and E. P. Xing. Quasi real-time summarization for consumer videos. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2513–2520, 2014.

[34] S. Zhao, H. Yao, X. Sun, X. Jiang, and P. Xu. Flexible presentation of videos based on affective content analysis. In *Proceedings of International Conference on Multimedia Modeling*, pages 368–379, 2013.