CLS: A Cross-user Learning based System for Improving QoE in 360-degree Video Adaptive Streaming

Lan Xie
Institute of Computer Science & Technology, Peking University / Beijing Hulu Software Technology Development Co., LTD
Beijing, China
xielan@pku.edu.cn

Xinggong Zhang*
Institute of Computer Science & Technology, Peking University / Cooperative Medianet Innovation Center
Beijing, China
zhangxg@pku.edu.cn

Zongming Guo
Institute of Computer Science & Technology, Peking University / Cooperative Medianet Innovation Center
Beijing, China
guozongming@pku.edu.cn

ABSTRACT

Viewport adaptive streaming is emerging as a promising way to deliver high quality 360-degree video. It is still a critical issue to predict user's viewpoint and deliver partial video within the viewport. Current widely-used motion-based or content-saliency methods have low precision, especially for long-term prediction. In this paper, benefiting from data-driven learning, we propose a Cross-user Learning based System (CLS) to improve the precision of viewport prediction. Since users have similar region-of-interest (ROI) when watching a same video, it is possible to exploit cross-users' ROI behavior to predict viewport. We use a machine learning algorithm to group users according to historical fixations, and predict the viewing probability by the class. Additionally, we present a QoE-driven rate allocation to minimize the expected streaming distortion under bandwidth constraint, and give a Multiple-Choice Knapsack solution. Experiments demonstrate that CLS provides 2dB quality improvement than full-image streaming and 1.5 dB quality improvement than linear regression (LR) method. On average, the precision of viewpoint prediction improve 15% compared with LR.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Human-centered computing** → *Virtual reality*;

KEYWORDS

360-degree video, tile-based adaptive streaming, viewport prediction, QoE-driven rate allocation

ACM Reference Format:
Lan Xie, Xinggong Zhang, and Zongming Guo. 2018. CLS: A Cross-user Learning based System for Improving QoE in 360-degree Video Adaptive Streaming. In *Proceedings of 2018 ACM Multimedia Conference (MM'18).* ACM, Seoul, Republic of Korea, 9 pages. https://doi.org/10.1145/3240508.3240556

*Dr. Xingong Zhang is the corresponding author.

MM'18, October 2018, Seoul, Republic of Korea

ACM ISBN 978-1-4503-5665-7/18/10…$15.00
https://doi.org/10.1145/3240508.3240556

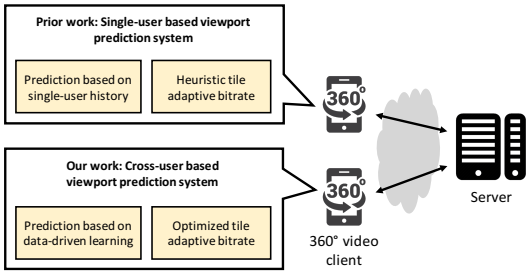


Figure 1: Design Comparison

1 INTRODUCTION

In recent years, many video service providers roll out 360-degree video which provide immersive experience to users [2]. While consuming 360-degree video, users can change their viewpoint, resulting in an interactive experience than consuming traditional video with a fixed viewing direction. However, 360-degree video's high resolution and bitrates demand hinder their wide spread over the Internet.

The streaming of 360-degree video is currently deployed in a naive way by simply streaming the entire 360-degree view in constant quality. However, only a portion of the video is viewed by the user at a specific time. As a consequence, transmitting entire 360-degree view results in inevitable waste of bandwidth and computational resources. Due to the prevalently use of HTTP-based adaptive streaming (standardized as DASH [19]), *viewport-adaptive streaming* [8] is regarded as a promising way to deliver 360-degree video through the Internet. One realization is tile-based streaming framework [4, 10, 13, 16, 18, 21]. In tile-based streaming, each temporal video segment is composed by several spatial tiles which can be independently encoded/decoded. It performs in such way that high quality is preserved within the tiles cover user's viewport while other tiles are delivered in low quality.

In video streaming, the client needs to buffer some amount of video to ensure continuous playback. Therefore, existing methods typically suggest to pre-fetch video segments by predicting the user's future viewport. The viewport prediction algorithms can be categorized into two classes: 1) single-user based algorithms [12, 13, 21] and 2) content-based algorithms [6]. However, they have key limitations:

- Single-user based algorithms only consider the single user's head motion and is unaware of the video content. As a result, the viewport prediction could easily be biased when predicting long-term future (e.g predicting user's viewport in future 5 second) [13]. Moreover, predictions could be inaccurate if the video content has large variation.
- Content-based algorithms apply traditional saliency or object detection algorithms on 360-degree videos to find ROI on content. However, these algorithms are high-computational and have relatively poor accuracy. This is because predicting ROI on 360-degree video is inherently different and more challenging compared to planer video.

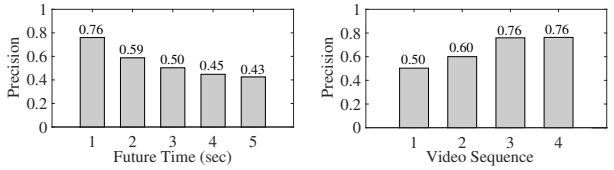
To address the above concerns, our key intuition is to leverage the advantage of data-driven analysis of cross-user behavior and machine learning techniques to determine the viewing probabilities of tiles. We first conduct user study from user behavior. Our user study indicates that 1) most users are drawn to similar ROI on 360-degree content. 2) There may have multiple ROI in a video.

Building on the above observations, we have designed a Cross-user Learning based System (CLS) for viewport-adaptive 360-degree video streaming. On the server side, it first do *user fixation clustering* to group users with similar viewing behavior. The users in same class watch similar video content. Then, the server computes the viewing probabilities of tiles for each class. For a new session, at each adaption step, the client first predicts the user's class according to this user's history fixations and obtain the corresponding viewing probabilities of tiles. Then, a QoE-driven rate allocation algorithm computes the best quality level for each tile considering network condition and object quality.

While details are presented in the paper, some highlights for our contributions include the followings:

- A user study under a real 360-degree video viewing behavior dataset is conducted. Benefiting from data-driven analysis, we find some key observations and give explanations.
- We propose a cross-user learning based viewport prediction algorithm. It can grab effective content-related information to some extent without using pixel-wise analyzing.
- We present a QoE-driven rate allocation problem to minimize the expectation of distortion under the bandwidth constraint which can be solved as a Multiple-Choice Knapsack problem.
- Extensive experiments are carried out on real-world Internet and user head movement trajectories. The results demonstrate that CLS achieves the highest video quality compared with the state-of-art methods.

The rest of this paper is organized as follows. Section II surveys related works on 360-degree video streaming and analysis the key limitations of these methods. In Section III, we conduct fixation user study and analysis the result. Then, based on the key observations in user study, we present the design of CLS in Section IV. Performance evaluation and comparison are presented in Section V. Finally, Section VI concludes the paper and outlines future directions.



(a) Precision drops along with future prediction time. (b) Precision varies among different content.

Figure 2: Limitations of single-user based viewport prediction.

2 BACKGROUND AND RELATED WORK

2.1 Tile-based 360-degree Streaming

In traditional HTTP-based adaptive streaming, a video is temporally partitioned into segments. To support viewport-adaptive streaming, video segments are further spatially divided into tiles, so that each temporal segment is composed by several spatial tiles. Since the client needs to buffer some amount of video to ensure continuous playback, it is necessary to pre-fetch video segments according to the result of viewport prediction. Viewport prediction can decide which tiles cover the user's future viewport in an absolute manner [13] or in a probability manner [21]. Then, rate allocation is responsible to select the quality level of tiles considering the viewport prediction and the estimated available bandwidth. As video segments are downloaded, the 360-degree video is rendered onto the screen of Head Mounted Display (HMD) using graphic engine.

Apart from tile-based streaming, Scalable Video Coding (SVC) based streaming [11] and asymmetric projection based streaming are the alternative strategies to achieve viewport adaptivity. The former can improve the video quality in viewport by requesting enhancement layers. The later uses asymmetric projection, e.g. Truncated Pyramid Projection [7], where the quality is decreased apart from the viewpoint.

2.2 Viewport Prediction

Currently, the viewport prediction algorithms can be categorized into two classes: single-user based and content based. Single-user based algorithms predict the future viewport according to the user's history fixations. Qian [13] and Stefano [12] use Linear Regression model to predict viewport. Lan [21] proposes a probabilistic model considering the prediction error follows Gaussian Distribution.

The first problem of existing viewport prediction algorithms is that these algorithms only consider the single-user's historical motion and consider it as a linear motion. As a result, the viewport prediction accuracy can be easily biased when predicting long-term future. Fig. 2(a) shows a trace-driven evaluation to highlight the prediction precision under different prediction time. We consider the linear regression model to evaluate. It uses history fixations in one second to predict the user's viewport in future 1 to 5 seconds. We see that the prediction precision drops heavily when predicting the viewport in future 2 seconds .

Another problem is that these algorithms are unaware of the video content. Hence, predictions could be inaccurate if the video content has large variation (e.g. motion-fast video, scene cut). To evaluate the impact of different kinds of content, we consider four videos in the dataset [20]. The former two videos contain motion-fast scene while the later two videos are relatively static. Fig. 2(b)

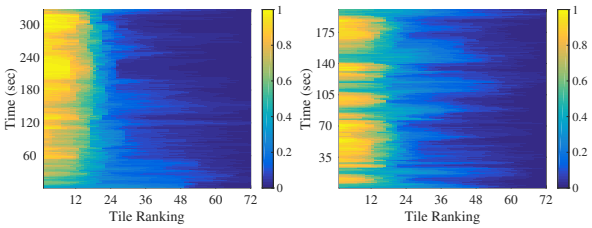


Figure 3: Heapmaps of the collective visibility.

shows the viewport prediction precision of these videos. We can conclude that the precision of single-user based algorithm highly related to video content.

For the content based prediction algorithm [6], the authors use saliency detection [3] and neural network to understand the ROI on content. However, predicting ROI on 360-degree video is inherently different and more challenging compared to planer video, since the 360-degree video is omnidirectional. Besides, it can not meet the real-time requirements in video streaming. Instead, we dig into the viewing behavior across users to understand video content.

3 FIXATION USER STUDY

With the increasing popularity of 360-degree videos, understanding user viewing behaviors in virtual environment is important for improving user experience. We conduct a user study to systematically analysis the viewing behavior of users when watching 360-degree videos.

3.1 Setup

Our user study is conducted on the VR dataset [20]. It contains 18 videos which are selected to represent the most popular video categories of current 360-degree contents and cover a wide range of scenarios, e.g. performance, sport, landscape, etc. 48 participants are involved in the experiments and their head motions are recorded during the video playback, including rotations and positions.

The primary task of the user study is to report the viewing behavior on the global view of all users. In tile-based video streaming, the adaptation granularity is tile. In this work, we fix the tiling setting and focus on the user viewing behavior. The same as [21], each video segment is divided into 6 rows and 12 columns (72 tiles). The viewport is considered with a 90-degree field of view (FOV) which is a common setting of HMD.

3.2 Methodology

The head movement is tracked on Unity 3D platform, in which the unit quaternion is recorded. The unit quaternion represents the rotations of an objects in 3D space. A unit vector (x, y, z) represents which direction a participant is looking at, i.e. fixation, can be calculated from the unit quaternion [20]. Then, the viewport has a limited FOV with center (x, y, z) and is modeled as a plane segment tangential to the sphere.

At a specific time, the *tile visibility vector* of a user u is defined as $\boldsymbol{v_u}$ which is a 0-1 vector. Specifically, $v_{u,i} = 1$ represents that the i-th tile is viewed by user u, otherwise the tile is not viewed by the user. Forward Projection [22] is used to calculate the tile visibility vector. To see the viewing behavior on the global view of all users,

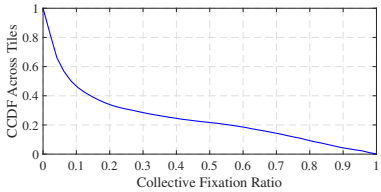
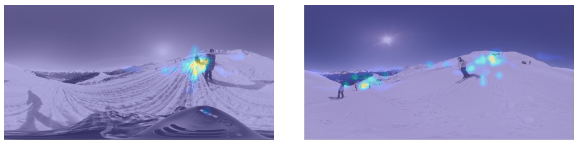


Figure 4: CCDF of collective visibility.



(a) Frame 1: One ROI in the content (b) Frame 2: Multiple ROIs in the content

Figure 5: User fixations on two video frames. Yellow region means high density of user fixations while blue region mean low density of user fixations.

we define the *collective visibility* of a tile, i.e. $\bar{v}_i$, as the fraction of users who view the tile. It is calculated as $\bar{v}_i = \frac{\sum_{u \in U} v_{u,i}}{|U|}$, where U is the set of all users.

3.3 Collective Tile Visibility Result

Fig. 3 shows the collective visibility of tiles of two videos. The row is the video playback time and the column is the tile index which is ranked according to the collective visibility. During 200 to 240 seconds of the first video, 12 tiles have a collective fixation of over 0.9. In other words, 90% of the users watch these tiles during this time. The second heatmap in Fig. 3 is calculated from a motion-fast video. We can still observe similar results. Fig. 4 shows the Complementary Cumulative Distribution Function (CCDF) of collective visibility of the first video. Each point represents the percentage of tiles with at least a certain collective visibility value. For example, the graph shows that only 10% tiles have a collective visibility higher than 0.8 and 50% tiles have a collective visibility less than 0.1. It indicates that 10% tiles are watched by 80% users and 50% of the tiles are viewed by less than 10% of the users. Therefore, our key takeway is: several tiles have high collective visibility and there is a significant number of regions that are relatively unimportant to the users.

3.4 Fixation Clustering Result

We have observed that users have similar ROI when watching same 360-degree video (§3.3). Obviously, if all users watch similar content, the server can just find these tiles according to the collective visibility and stream these tiles to the client at high quality level.

However, users have their own preference when watching 360-degree videos. This phenomenon is also ubiquitous in 360-degree video. To see whether users have different preferences when watching 360-degree videos, we conduct experiments to study the clustering of users fixations. Fig. 5 show the user fixations on two frame of video skiing. Yellow region means high density of user fixations while blue region means low density of user fixations. In Fig. 5(a), a man is skiing and dragged by a sledge. Therefore, there is one obvious ROI and almost all users focus on that region. In Fig. 5(b), a man is skiing down from the high hill and several people are skiing

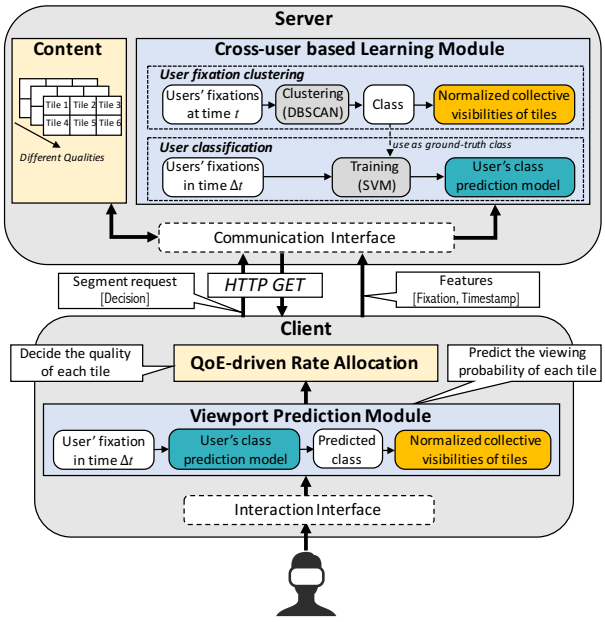


Figure 6: Diagram of CLS architecture.

down to the low hill. Therefore, the user fixations are separated into two groups according to their own preference on content.

Consider the observation from §3.3, all users share the same tiles' weights, i.e. tile viewing probabilities, may result in quality decrease in the multi-ROIs case. For example, in the situation shown in Fig. 5(b), the tiles cover the two ROIs will be assigned high weights than other tiles. However, the tiles in the left ROI are unimportant to the users in the right cluster. If we can predict which cluster a user belongs to, the client can assign high bitrate to the tiles with high collective visibility in that cluster. Hence, the video quality can be improved.

4 SYSTEM DESIGN

To breakthrough the limitations of using single-user based viewport prediction in 360-degree streaming systems, we observe a key ROI-aware insight from cross-user fixation analysis that enables us to address the challenges in viewport prediction. In this section, we present the design of CLS which is built upon this insight.

4.1 System Overview

The system architecture is shown in Fig. 6. To achieve cross-user based learning, the client sends the measured features, including user fixation and the corresponding video playback timestamp to the server. At the server, the Cross-user Learning Module is responsible for supporting viewport prediction. Specifically, in order to find ROI(s) in each video segment, the user fixations are used to group users into class(es). For users in same class, since they watch similar content, they have same viewing probabilities of tiles. The detail about the CLS algorithm is shown in §5.1.

When a new session comes in, it first requests for video manifest file (e.g. MPD) to obtain the information about the video, including the 1) pre-learned models and viewing probability for supporting

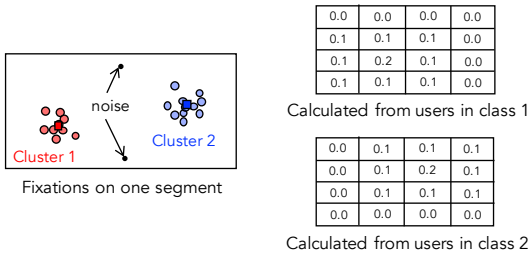


Figure 7: Illustration of cross-user learning based viewport prediction.

viewport prediction; 2) quality and bitrate for each tile for supporting rate allocation. At each adaptation step, the client first predicts the user's class to estimate the user's preference on the requested video segment. Then, the client obtains the viewing probability that belongs to the class. At last, the client determines the quality level for each tile according to the QoE-driven rate allocation. The details about the QoE-driven rate allocation are presented in §5.3.

The new session also reports the measured features to the server. When the server receives the feedback data, it again updates the Cross-user Learning Module using the new features.

5 CLS ALGORITHM

5.1 Cross-user based Learning

At the server side, the Cross-user Learning based Module is used to support client side viewport prediction. It is decoupled into two parts: a *user fixation clustering* logic to partition users into class(es) according to their preference on content and a *user classification* logic to predict a user's future class, i.e. which ROI he/she will interest in the future. We implement the Cross-user Learning based Module with Scikit-learn [1] which is a machine learning library in Python.

5.1.1 User Fixation Clustering. The user fixation clustering is done by server using old sessions' feedback data. We denote that the fixation coordinates of user u at specific time t as (x_u, y_u, z_u), a unit vector represents which direction a user is looking at. We assume that users maintain his/her fixation within a small time interval. Therefore, the granularity of user clustering is the same as segment duration, e.g. one second. Since fixation coordinate is spatial data with noise and the number of cluster(s) can vary in each clustering interval, we use the density based clustering algorithm DBSCAN [5] to group users. Given the set of users' fixations on one video segment, DBSCAN can group together the users' fixation that are closely packed together, i.e. fixations that have enough nearby neighbors are grouped together as a class. It starts by deciding a core fixation and the fixations that are reachable[1] from the core fixation are grouped into the same class. The fixation is a noise point if it is non-reachable by the points in each class. We use Euclidean distance as the distance function. For example, as shown in Fig. 7, each point is the user fixation on one segment of 360-degree video. The users can be clustered into two classes according to their fixations. It also reveals that there are two ROIs in this video

[1]Two points are reachable if 1) the distance between the two points is less than ε or 2) there is a path between the two points satisfies that the distance of any adjacent points is less than ε.

segment. The clustering result of user u is denoted as $c_u \in [0, C]$, where C is the number of cluster(s). Specifically, $c_u = 0$ represents the user is treated as noise.

After user clustering, the viewing probability of each tile is determined within each class of users. Fig. 7 illustrates the viewing probability of two classes in a 4×4 tilling manner as an example. Specifically, it is calculated from collective visibility. We denote the set of users in a specific class α as U_α. Therefore, the collective visibility for i-th tile for class α is calculated as:

$$\bar{v}_{\alpha,i} = \frac{\sum_{u \in U_\alpha} v_{u,i}}{|U_\alpha|}. \tag{1}$$

We denote the viewing probability of i-th tile for class α as $p_{\alpha,i}$. It is calculated as the normalized result of the corresponding collective visibility:

$$p_{\alpha,i} = \frac{\bar{v}_{\alpha,i}}{\sum_i \bar{v}_{\alpha,i}}. \tag{2}$$

5.1.2 User Classification. For the client, it will pre-fetch video segments from sever according to the viewport prediction result. However, the client is unconscious about the user's future class, which is crucial in viewport prediction. Hence, the client needs to predict the user's class to obtain the corresponding viewing probabilities of tiles. To support this, the server trains model using machine learning. Specifically, for predicting user's class at video playback time t, it uses the user clustering result at playback time t as ground-truth and the user fixations in time window $\Delta t = [t - B_{\max} - \delta, t - B_{\max}]$ as features, where $B_{\max}$ is the buffer length and δ is the window size in time. We use $(t - B_{\max})$ as the upper bound of the window since the maximum length of video that client can store in the playback buffer is $B_{\max}$. Thus, the maximum prediction time horizon is $B_{\max}$. We use Support Vector Machine (SVM) [9] as the class prediction method. Noting that other prediction methods can be used in our task as well, SVM is a a widely used method with enough high performance and low computation.

5.2 Viewport Prediction

At the beginning of 360-degree video session, the client first downloads the pre-trained model and the viewing probabilities of tiles for each class from the server to support the viewport prediction during video playback. In each adaptation interval, the client first predicts the user's future class using the historical fixations as features. Then, the client obtains the viewing probabilities of tiles correspond to the predicted user class.

5.3 QoE-driven Rate Adaptation

To provide high QoE, we propose the rate allocation of tiles as a QoE-driven optimization problem. We define the number of tiles in one segment as N and the number of bitrate versions as M. At each adaptation step, the bitrate of i-th tile at j-th quality level has a bitrate $r_{i,j}$ and a spherical quality distortion $d_{i,j}$ which can be obtained from manifest files. We define the optimal quality level for i-th tile as l_i. To be simplified, we use p_i to represent the viewing probability of a specific predicted class. Therefore, at each adaptation step, the client needs to solve the following optimization problem to obtain the optimal quality level for each tile:

$$\begin{aligned} \min_{l_i \in [1,M], \forall i} \quad & \sum_{i=1}^{N} p_i \cdot d_{i,l_i} \\ \text{s.t.} \quad & \sum_{i=1}^{N} r_{i,l_i} \le BW. \end{aligned} \tag{3}$$

The constraint in the optimization problem restricts the total bitrate of tiles. To avoid playback interrupts, we set a transmission bitrate budget BW which is calculated from buffer based rate adaptation algorithm [21]. Besides, to avoid blank block in user's viewport, the tiles will be assigned at least the lowest quality level.

This optimization problem can be solved as a Multiple-Choice Knapsack problem [17]. A brute force search which exhaustively evaluates all combinations guarantees an optimal solution. However, the computational complexity is $O(N^M)$. To reduce the computation time, we use algorithm 1 to solve the problem where the computational complexity is $O(BW \cdot N)$.

Algorithm 1 QoE-driven Rate Allocation Algorithm.

Input: Throughput bound, BW; Number of tiles, N; Number of rates, M; Rate set, $\{r_{i,j}\}$; Tile weights, $\{p_i\}$; Distortion set, $\{d_{i,j}\}$;
Output: Allocation rate levels set, $\{l_i\}$;

$\forall i$, appoints $l_i = 1$;
Update the remaining throughput $BW' = BW - \sum_i r_{i,1}$;
Update the rate set by $\{r'_{i,j} = r_{i,j} - r_{i,1}\}$;
Update the distortion set by $\{d'_{i,j} = p_i(d_{i,1} - d_{i,j})\}$;
Initialize knapsack revenue table $\mathcal{K} \in \mathbb{R}^{(N+1)\times BW'}$ by 0;
Construct the prefix table $\mathcal{P} \in \mathbb{Z}^{(N+1)\times BW'}$;
for i from 1 to N **do**
  for $bw \in [0, BW']$ **do**
    $\mathcal{K}_{i,bw} = \max_{j\in[1,M]}\{\mathcal{K}_{i-1,bw-r'_{i-1,j}} + d'_{i-1,j}\}$;
    $\mathcal{P}_{i,bw} = \arg\max_{j\in[1,M]}\{\mathcal{P}_{i-1,bw-r_{i-1,j}} + d'_{i-1,j}\}$;
  end for
end for
Find $B\hat{W} = \arg\max_{bw\in[0,BW']}\{\mathcal{K}_{N,bw}\}$;
for i from N to 1 **do**
  $l_i = \mathcal{P}_{i,B\hat{W}}$;
  $B\hat{W} = B\hat{W} - r'_{i,l_i}$;
end for
return $\{l_i\}$

6 PERFORMANCE EVALUATION

To evaluate the performance of CLS, we carry out extensive real-world Internet experiments using user head movement trajectories.

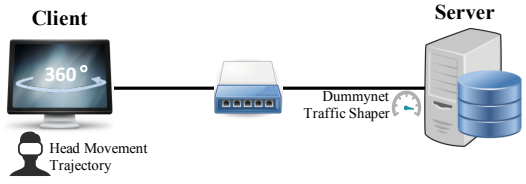
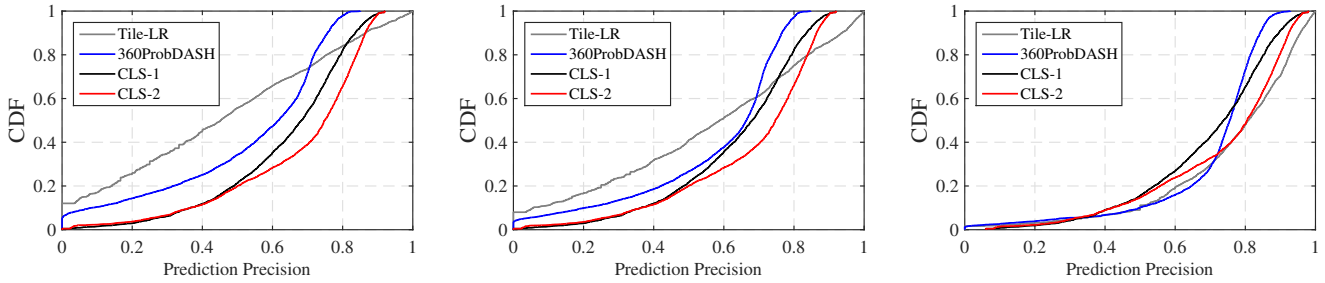
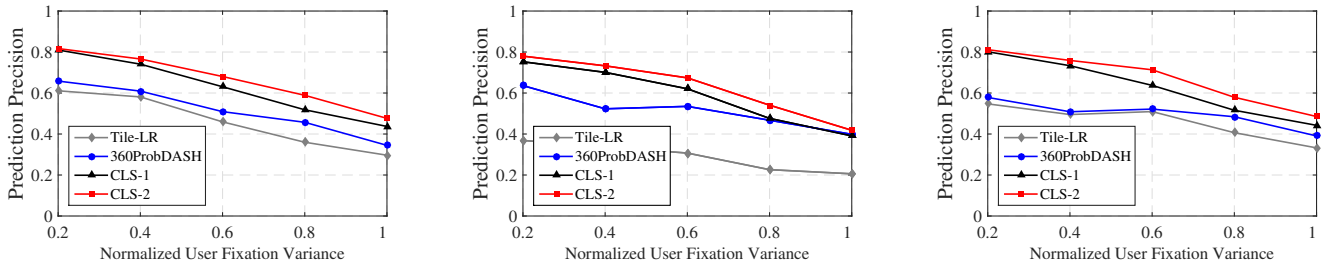


Figure 8: Network topology



(a) Prediction time horizon = 5 sec (b) Prediction time horizon = 3 sec (c) Prediction time horizon = 1 sec

Figure 9: CDF of viewport prediction precision of different prediction time horizons.



(a) Video 1 (b) Video 2 (c) Video 3

Figure 10: Viewport prediction of three videos under different user fixation variance.

6.1 Setup

Table 1: Video Information

No.	Content	Length	Category
1	Conan360°- Sandwich	2'44"	Performance
2	Freestyle Skiing	3'21"	Sport
3	Google Spotlight - HELP	4'53"	Film

Fig. 8 shows the network topology in the experiment, which consists of a client and a server. We also throttle the bandwidth using Dummynet [15] in order to emulate different bandwidth settings. Head movement trajectories are embedded into the client to imitate the head motion when user watches 360-degree videos.

In the experiments, we choose three videos in different types from the VR dataset [20]. The information of these video is shown in Table 1. We set the duration of one video segment as 1 second. We adopt the 6×12 (rows $\times$ columns) tiling pattern for each video segment, thus the number of tiles is 72 ($N = 72$). To generate different quality videos, we use quantization parameter (QP) ranging from 22 to 42 in steps of five leading to five different bitrate versions. The perceptual quality distortion of each tile is calculated using Spherical MSE which is the value to further calculate viewport PSNR (V-PSNR) [22]. The length of playback buffer is set to 5 seconds. In the Cross-user based Learning Module, we randomly select 10 percent ($48 \times 0.1 \approx 5$) traces for each video as validation set and 43 traces as training set.

To validate the efficiency of cross-user based viewport prediction in 360-degree video streaming, we select three typical streaming methods as the comparisons:

- MONO: This approach is monolithic streaming. The naive way by streaming the entire 360-degree scene in constant quality without exploiting and optimizing the quality for the user's viewport.
- Tile-LR [13]: This tile-based streaming uses Linear Regression to predict future viewport.
- 360ProbDASH [21]: This approach use a probabilistic model to predict viewing probabilities of tiles. It considers that the prediction error follows a Gaussian Distribution.

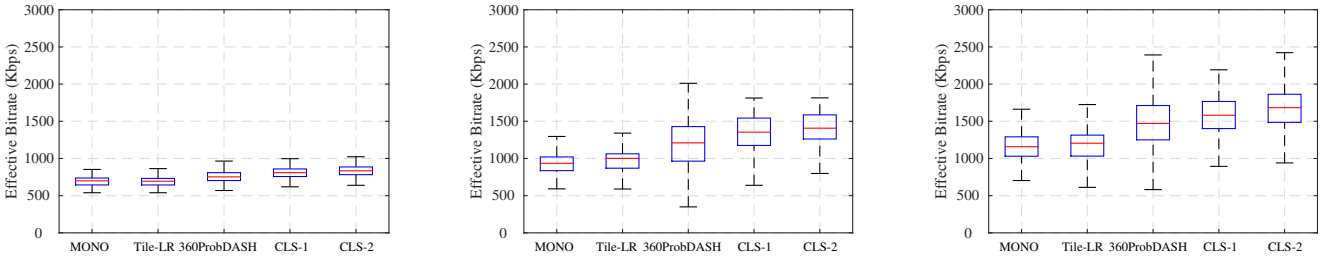
Moreover, we also test the performance of our proposed method without user classification which is referred to as CLS-1. This way, it is possible to clearly identify the gains of user classification. The proposed method with user classification is referred to as CLS-2. For fair comparison, the buffer based rate adaptation [21] is used for all methods. The proposed QoE-driven rate allocation is used in each of the tile-based methods. The video bitrate for MONO method is chosen at the highest bitrate less than the rate adaptation result.

We evaluate the effectiveness of our proposed method by comparing with other methods in viewport prediction precision, video quality over fixed bandwidth network and real-world Internet.

6.2 Viewport Prediction

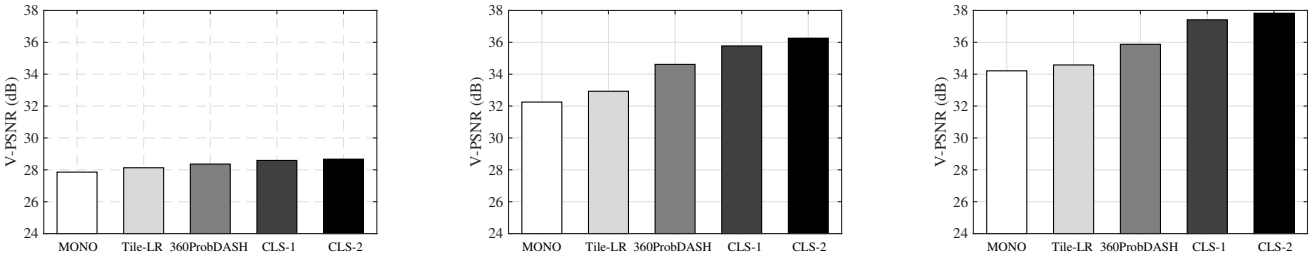
We first evaluate the viewport prediction precision of these methods. Since the viewing probabilities of tiles are calculated in a normalized manner, i.e. $\sum_{i=1}^{N} p_i = 1$, the precision of viewport prediction is calculated as $\sum_{i=1}^{N} \min\{p_i, g_i\}$ where g_i is the normalized ground-truth.

Recall that the client needs to pre-fetch some video segments to prevent playback interruption. In most adaptation steps, the client requests the segment to fill-up playback buffer. In these cases, the viewport prediction time horizon equals to the buffer length (5 seconds). Fig. 9(a) shows the CDF of viewport prediction precision



(a) Bandwidth = 2000Kbps (b) Bandwidth = 3000Kbps (c) Bandwidth = 4000Kbps

Figure 11: Effective bitrate under different fixed bandwidth situations.



(a) Bandwidth = 2000Kbps (b) Bandwidth = 3000Kbps (c) Bandwidth = 4000Kbps

Figure 12: V-PSNR under different fixed bandwidth situations.

when predicting user's viewport in future 5 seconds. On an average, the CLS-2 achieves 75% prediction precision while CLS-1, 360ProbDASH and Tile-LR achieve 70%, 60% and 45% prediction precision. At the 80th percentile, CLS-2 improves 5%, 10% and 15% compared to CLS-1, Tile-LR and 360ProbDASH respectively. In rare cases, when the playback buffer contains less video content, the client should predict user's viewport in the near future to download the video segment. Fig. 9(b) and 9(c) show the the CDF of viewport prediction precision when predicting user's viewport in future 3 seconds and 1 second. At the 80th percentile, CLS-2 can achieve 85% and 90% prediction precision in prediction time horizon of 3 seconds and 1 second respectively.

According to Fig. 9, we observe that the viewport prediction precision decreases with the increase of prediction time. Such degradation is noticeable in linear regression method, i.e. gray line. The proposed method presents a high robustness in prediction time against with others methods. This confirms that considering cross-user behavior provides high viewport prediction than only using single-user historical fixations.

We use the variance of user fixations to represent the fixation centrality. A small value represents high similarity of user fixations which further implies noticeable ROI in video content. Fig. 10 shows the viewport prediction of three videos under different user fixation variances. Each point on the figure represents the average prediction precision of predicting the user's viewport in video segments that with same user fixation variance. Generally, the viewport prediction precision decreases with the increase of user fixation variance. It means that the viewport prediction precision has a strong correlation with content, e.g. the viewport in video segment with noticeable ROI is easier to predict than the segment without noticeable ROI.

Fig. 10 illustrates that CLS-2 outperforms other prediction algorithms in all cases. In Fig. 10(a) and Fig. 10(c), CLS-2 achieves 80% prediction precision when the normalized user fixation variance is less than 0.2. Since video 1 and video 3 are motion-less, from the statistics, nearly 54% video segments have a user fixation variance less than 0.2 (not shown due to space limitation). Even under the highest user fixation variance, CLS-2 can achieve 50% prediction precision. However, video 2 is motion-fast, we can see a large prediction precision drop in Tile-LR while CLS-2 still can preform high prediction precision.

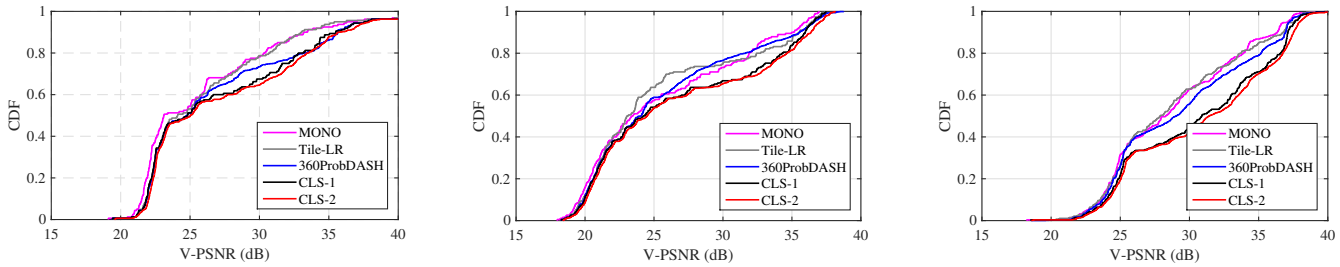
6.3 Video Quality under Fixed Bandwidth

In 360-degree video streaming, viewport prediction influences the bitrate allocation of tiles and perceptual quality of content in user's viewport. Since only a portion of the video is viewed by the user, the perceptual quality is decided by the tiles cover the user's viewport. We define *effective bitrate* as the sum of video bitrate of tiles cover user's viewport. Besides, we use V-PSNR [22] to directly represent the video quality inside the user's viewport. To evaluate the performance in video quality of different viewport prediction algorithms, we first conduct Internet experiments under fixed bandwidth {2000, 3000, 4000} Kbps.

Fig. 11 shows boxplot of effective bitrate of different methods during streaming sessions for video 1. The minimum value of whisker (dashed line) represents the lowest effective bitrate, the red line is the median of effective bitrate, the maximal value of whisker is the highest effective bitrate, the box contain 50% values of effective bitrate. When the bandwidth is low, as shown in Fig. 11(a), the median effective bitrate is around 700kbps for the 5 methods. Since in most adaptation steps, the client chooses the lowest quality for each tile. When the capacity is adequate, as shown in Fig. 11(c),

Table 2: Performance Summary

Algorithm	Video 1			Video 2			Video 3		
	*s*Rate (Kbps)	*e*Rate (Kbps)	V-PSNR (dB)	*s*Rate (Kbps)	*e*Rate (Kbps)	V-PSNR (dB)	*s*Rate (Kbps)	*e*Rate (Kbps)	V-PSNR (dB)
MONO	2418.1	808.2	27.16	3458.1	1015.5	25.54	3312.5	1072.5	28.74
Tile-LR [13]	2442.5	868.2	27.44	3513.9	1035.6	25.69	3306.6	1099.4	28.85
360ProbDASH [21]	2333.1	958.1	28.12	3311.7	1177.9	25.86	3215.3	1186.1	29.39
CLS-1	2431.8	1044.3	28.40	3465.4	1396.7	26.54	3315.8	1360.3	30.61
CLS-2	2435.5	**1174.3**	**29.10**	3473.1	**1468.3**	**27.09**	3319.6	**1443.9**	**31.12**



(a) Video 1 (b) Video 2 (c) Video 3

Figure 13: CDF of V-PSNR of three videos.

the median of effective bitrate for CLS-2 is 1650Kbps which is the highest median value among all methods. Besisde, the box area of CLS-2 ranges in [1500,1900] Kbps while CLS-1, 360ProbDASH, Tile-LR and MONO range in [1450,1750] Kbps, [1250,1740] Kbps, [1010,1320] Kbps and [1010,1300] Kbps respectively.

Fig. 12 shows the average V-PSNR of different methods during streaming sessions for video 1. Fig. 12(b) shows that Tile-LR, 360ProbDASH, CLS-1 and CLS-2 enhance the V-PSNR by 0.9dB, 2.1dB, 3.7dB and 4dB than MONO in 3000Kbps bandwidth. This indicates that viewport prediction can improve the perceptual quality for 360-degree video and the prediction algorithm is essential. Overall, crossUser-2 can improve the V-PSNR by 1.5dB, 4dB and 3.8dB than MONO in 2000Kbps, 3000Kbps and 4000Kbps bandwidth respectively.

6.4 Video Quality under Real-world Internet

To further evaluate the performance under severe network conditions, we conduct a series of experiments under real-world Internet. To make the experiment repeatable, we choose a bandwidth trace from HSDPA dataset [14]. The videos and test user head movement trajectories are same with experiment setup.

Fig. 13 shows the CDF of V-PSNR of different methods on three videos. As can be seen, all methods have similar result when V-PSNR is lower than 25 dB. We anticipate these results because these adaptation steps suffer from low bandwidth condition, which is consistent with the result in Sec 6.3. However, CLS-2 can significantly improve the V-PSNR when the bandwidth is adequate. For example, in Fig. 13(b), in 80th percentile, CLS-2 improves the V-PSNR by nearly 3 dB than other methods.

Table 2 summaries the performance of different methods on segment bitrate (*s*Rate), effective bitrate (*e*Rate) and V-PSNR. Since only the tiles that cover user's viewport are actually viewed by user, effective bitrate is always lower than segment bitrate. We can see all tile-based methods achieve higher effective bitrate and V-PSNR than MONO. This is because tile-based viewport adaptive streaming can optimize the quality of tiles, while traditional monolithic streaming assigns same quality level to tiles. Meanwhile, CLS-2 outperforms other methods in effective bitrate and V-PSNR. This is due to the fact that CLS-2 provides an accurate viewport prediction result, which eventually leads to a highest perceptual quality.

As can be seen in Table 2, video 2 has a lower V-PSNR than other videos. This is because video 2 is a motion-fast video which has a lower compression efficiency than the motion-low videos. This implies that the perceptual video quality is influenced by video content. On an average, CLS-2 can enhance the V-PSNR by 2dB, 1.8dB and 1.3dB than MONO, Tile-LR and 360ProbDASH over the three videos respectively.

7 CONCLUSION

Viewport adaptive streaming is emerging as a promising way to deliver high quality 360-degree video. In this paper, we conduct user study under a real VR dataset. Our user study indicates that 1) most users are drawn to similar ROI on 360-degree content. 2) the number of ROIs varies across video content. Based on the above insights, we propose a Cross-user Learning based System (CLS) to improve the viewport prediction precision for tile-based 360-degree video streaming. Specifically, we can find region-of-interest (ROI) from cross-user behavior analysis and predict the user's preference on content using machine learning. Then, we present a QoE-driven rate allocation problem to minimize the expectation of distortion under the bandwidth constraint which can be solved as a Multiple-Choice Knapsack problem. Experiments demonstrate that CLS outperforms the state-of-art tile-based streaming methods.

ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundation of China under contract No. 61471009 and Culture Development Funding under Grant No.2016-288.

REFERENCES

[1] *Scikit-learn.* http://scikit-learn.org/stable/

[2] *YouTube live in 360 degrees encoder settings.* https://support.google.com/youtube/answer/6396222
[3] Ali Borji, Ming-Ming Cheng, Qibin Hou, Huaizu Jiang, and Jia Li. 2014. Salient object detection: A survey. *arXiv preprint arXiv:1411.5878* (2014).
[4] Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. 2017. Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming. *in Proc. of ACM Multimedia (MM)* (2017).
[5] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, and others. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *Kdd*, Vol. 96. 226–231.
[6] Ching-Ling Fan, Jean Lee, Wen-Chih Lo, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. 2017. Fixation Prediction for 360 Video Streaming to Head-Mounted Displays. (2017).
[7] M. Coban G. V. Auwera and M. Karczewicz. 2016. VR/360 Video Truncated Square Pyramid Geometry for OMAF. *ISO/IEC JTC1/SC29/WG11/M* (2016).
[8] Mario Graf, Christian Timmerer, and Christopher Mueller. 2017. Towards bandwidth efficient adaptive streaming of omnidirectional video over http: Design, implementation, and evaluation. In *Proceedings of the 8th ACM on Multimedia Systems Conference*. ACM, 261–271.
[9] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications* 13, 4 (1998), 18–28.
[10] Mohammad Hosseini and Viswanathan Swaminathan. 2016. Adaptive 360 VR video streaming based on MPEG-DASH SRD. In *Multimedia (ISM), 2016 IEEE International Symposium on*. IEEE, 407–408.
[11] Xing Liu, Qingyang Xiao, Vijay Gopalakrishnan, Bo Han, Feng Qian, and Matteo Varvello. 2017. 360 Innovations for Panoramic Video Streaming. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*. ACM, 50–56.
[12] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, and Filip De Turck. 2017. An HTTP/2-Based Adaptive Streaming Framework for 360 Virtual Reality Videos. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 306–314.
[13] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. 2016. Optimizing 360 video delivery over cellular networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. ACM, 1–6.
[14] Haakon Riiser, Tore Endestad, Paul Vigmostad, Carsten Griwodz, and Pål Halvorsen. 2012. Video streaming using a location-based bandwidth-lookup service for bitrate planning. *ACM Trans. Multimedia Comput. Commun. Appl (TOMCCAP)* 8, 3 (2012), 24.
[15] Luigi Rizzo. 1997. Dummynet: a simple approach to the evaluation of network protocols. *ACM SIGCOMM Computer Communication Review* 27, 1 (1997), 31–41.
[16] Patrice Rondao Alface, Maarten Aerts, Donny Tytgat, Sammy Lievens, Christoph Stevens, Nico Verzijp, and Jean-Francois Macq. 2017. 16K Cinematic VR Streaming. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1105–1112.
[17] Prabhakant Sinha and Andris A Zoltners. 1979. The multiple-choice knapsack problem. *Operations Research* 27, 3 (1979), 503–515.
[18] Kashyap Kammachi Sreedhar, Alireza Aminlou, Miska M Hannuksela, and Moncef Gabbouj. 2016. Viewport-Adaptive Encoding and Streaming of 360-Degree Video for Virtual Reality Applications. In *Multimedia (ISM), 2016 IEEE International Symposium on*. IEEE, 583–586.
[19] ISO/IEC JTC1/SC29/WG11 W13533. 2012. MPEG DASH: The Standard for Multimedia Streaming over the Internet.
[20] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference*. ACM, 193–198.
[21] Lan Xie, Zhimin Xu, Yixuan Ban, Xinggong Zhang, and Zongming Guo. 2017. 360ProbDASH: Improving QoE of 360 Video Streaming Using Tile-based HTTP Adaptive Streaming. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 315–323.
[22] M. Yu, H. Lakshman, and B. Girod. 2015. A Framework to Evaluate Omnidirectional Video Coding Schemes. In *2015 IEEE ISMAR*. 31–36.