The Frame Latency of Personalized Livestreaming Can Be Significantly Slowed Down by WiFi

Guoshun Nan[‡], Xiuquan Qiao[‡], Jiting Wang[*‖], Zeyan Li[*‖], Jiahao Bu[*‖], Changhua Pei[¶], Mengyu Zhou[†], Dan Pei[*‖§]
[‡]Beijing University of Posts and Telecommunications [*]Tsinghua University [†]Microsoft Research [¶]Alibaba Group
[‖]Beijing National Research Center for Information Science and Technology(BNRist)
nanguoshun@gmail.com, qiaoxq@bupt.edu.cn, {wangjt15,zy-li14,bjh16}@mails.tsinghua.edu.cn,
changhua.pch@alibaba-inc.com, mezho@microsoft.com, peidan@tsinghua.edu.cn

***Abstract*—The popular personalized livestreaming (PL) in China, arguably the largest PL market in the world, is more monetized than PL in US and hence demands much lower interactive latencies to ensure a good quality of user experience. However, our pilot experiment shows that the video frame latency, dominant component of PL's interactive latency, can be significantly slowed down by WiFi, the primary Internet access method for PL. Understanding and further improving the frame latency over WiFi, however, have difficulties in 1) measuring *end-to-end* latency; 2) parsing encrypted PL's traffic and 3) modeling complex relationships between WiFi radio factors and the latency. To tackle these challenges, we design and prototype Latency Doctor (LTDr), a practical system which aims to model and optimize PL's video frame latency over WiFi. We deploy LTDr in our campus and obtain several key observations based on 13.9M video frames extracted from 12K individual views on three leading PLs in China. We observe that 40% frame latencies over WiFi hop are more than 30ms, and channel utilization should be less than 64% for low latency. Then we build a predictive model based on the dataset using the machine learning methodologies. Two real cases show that the median frame latencies are decreased by LTDr from 130ms to 22ms, and 50ms to 12ms respectively over WiFi networks.**

I. Introduction

In the past few years, personalized livestreaming (PL) has become very popular around the world. Compared with traditional video services, it requires low latency to support high interactions between a PL's broadcaster and thousands of audiences. This interactive latency becomes one of the most important metrics for quality of user experience (QoE) of PL applications [1], [2] in addition to the well-known metrics such as playback buffering. Previous studies [1], [2] have helped us understand the characteristics and the performance of Periscope [3] in US. However, the landscape of PLs in China is totally different. PLs in China are more monetized in that broadcasting becomes a profession: broadcasters can make a living by actively interacting with viewers (*e.g.,* react to viewers' requests) and then receiving cash gifts from the viewers (see details in Section II). Hence, much lower interactive latency is required to meet viewer's demand compared with PL applications [3] in US. On the other hand, WiFi is the primary Internet access method and delivers 43% of total Internet traffic [4], while WiFi can be the major contributor of the *end-to-end* packet latency in densely populated areas where WiFi congestion and interference are high [5], [6]. In summary, compared to Periscope in US, much lower latency is demanded in China to ensure a good QoE, yet the wireless network connectivities are potentially much worse.

For above reasons, this paper aims to characterize and further reduce the latency of the personalized livestreaming over WiFi (PLoW for short) in China. To this end, we need to measure the *end-to-end* interactive latency of PLoW, more specifically, video frame latency. We also measure the WiFi radio factors such as channel utilization, queue length, *etc.* Then we study the impact of radio factors on PLoW's latency and build a predictive model. Such an understanding and a model can help WiFi protocol designers, WiFi network owners and designers, and PL applications to make the right decisions to mitigate, alleviate, or get around of WiFi's negative impacts on PL's latency. For example, suppose a PL App is experiencing long frame latency and it observes the current WiFi channel is congested, it can then switch to an alternative wireless access point (AP) with a less congested channel (see our implementation of this approach later in Section VI-B).

There exist three major challenges for achieving above goals. First, it is infeasible to measure the *end-to-end* frame latency for PLs in China for third parties like us. The timestamp for frame delivery can be measured on the client side on smart phones, while there is no timestamp for frame creation at the broadcaster side in any major PLs in China (different from Periscope [1]). To tackle this challenge, we opt to measure the "frame latency on the WiFi hop" instead, and use Open-Wrt based wireless AP as instruments to measure the timestamps when the frame is delivered to an AP. In fact, as will be shown in our testbed experiments in Section III, frame latency on the WiFi hop contributes to more than 50% of the *end-to-end* frame latency, for 25% of the video frames, thus it is worthwhile to single it out anyway. On the other hand, the AP instruments can naturally measure the WiFi radio factors on AP as done in [5]. It should be noted that the PL's frame latency at the WiFi hop is quite different from WiFi's per-packet latency, which can be simply measured by the *ping*

Dan Pei[§] is the corresponding author.

978-1-5386-6808-5/18/$31.00

command at a mobile client (illustrated in Fig. 2 and Fig. 4).

Second, to measure the frame latency at the WiFi hop, the video traffic captured on an AP should be parsed to obtain the arrival time of the first packet of a frame. However, the PL traffic on AP and commercial PL Apps are both encrypted, making it hard to measure the frame latency over WiFi. To tackle this second challenge, we take advantage of the unencrypted video API with RTMP [7] URLs, and modify a video player (ijkPlayer [8]) to directly request the RTMP URLs of PL broadcasts to mimic a real viewer, so that the packet receiving time can be recorded by the modified ijkPlayer and unencrypted RTMP streaming can be captured on an AP. Thus we can pinpoint the time points of the TCP packets that are formed into a video frame to calculate each frame latency at the WiFi hop offline based on the video traces.

Third, it is challenging to build simple yet accurate models to understand the impact of WiFi factors on PL latency and even build predictive models, given the potentially complex relationships among factors and between factors and PL performance. We use Random Forests/Decision Trees to tackle this challenge based on a large dataset collected on our testbed. We then verify their effectiveness on both residential and EWLAN 802.11 wireless networks.

Overall, we propose and implement Latency Doctor (LTDr), a system framework to measure and reduce the latency of PLoW. LTDr has been deployed on our university campus network in China for three months. The resulting dataset consists of 12 thousand RTMP broadcast views with 13.9 million video frames from Inke [9], Meipai [10] and Yi [11], which are three leading commercial PL platforms in China. We then characterize and quantify the complex relationships between WiFi radio factors and the WiFi hop frame latency. Finally, two practical applications are deployed to reduce the PL's frame latency over WiFi by model-based optimization.

This paper's major contributions are summarized as follows:

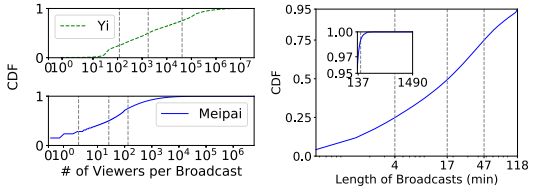
- We present the first study on PL in China, and find that PL characteristics in China are quite different from Periscope, *e.g.*, it is more monetized, and the amount of online broadcasters, viewers and views are much larger (Section II-A).
- To the best of our knowledge, we provide the first large-scale dataset consisting of video frame latency of PL over WiFi, which we plan to release to the public. The dataset is re-constructed from RTMP-based video traces that are collected by a modified media player rather than simulations or scripts. Furthermore, our WiFi testbed can be scaled by a carefully designed traffic generator to enrich the parameter space (Section IV).
- We present the first study on WiFi hop frame latencies for PL. The results show that 40% of the frame latencies are larger than 30ms, leading to a poor user experience for interactive streaming [12] (Section IV).
- We characterize the complex relationships between WiFi factors and the WiFi hop frame latency, and also quantify the significant impact of channel utilization and queue length on the quality performance (Section V).
- To the best of our knowledge, LTDr is the first systematic work that can help a wireless AP owner or operator to monitor the PL latency and prioritize the WiFi factors to improve the PL QoE. We build and deploy a predictive model based on our dataset using machine learning methodologies. Two real applications deployed in our campus show that median frame latencies are decreased from 130ms to 22ms, and 50ms to 12ms over residential and EWLAN WiFi networks, respectively.

II. Personalized Livestreaming in China

A. Application characteristics

We have crawled a dataset that consists of 0.91 million broadcasts from two leading PL platforms, *i.e.,* Meipai and Yi from 20th December 2016 to 10th February 2017. We focus on the interesting findings that show a difference from Periscope [1], [2]. The datails are described as follows.

1) PLs in China are more popular: As depicted in Fig. 1(a), the medians of viewer number per broadcast of Yi and Meipai are 2000 and 400, which are 200 and 40 times larger than Periscope's viewing figures[1], *i.e.,* 10 viewers.



(a) (b)

Fig. 1. Application characteristics. (a) Amount of viewers per broadcast, (b) Length of broadcasts for Meipai.

2) The lengths of broadcasts in China are much longer: As shown in Fig. 1(b), the 25th, 50th and 75th percentile values are 4min, 17min and 47min for Meipai, respectively, while they are 1min, 4min and 10min for Periscope [1]. It shows that 10% of broadcasts last more than 137 minutes.

3) PLs in China are more monetized: We observe that a broadcaster in China can make money from online users through talent shows, such as singing and telling jokes. Hence, much lower interactive latency is required for PL applications in China compared with those in the United States.

B. Interactive latency (ITL)

Fig. 2 breaks down the interactive latency (ITL) for a RTMP-based PL broadcast during which there is an interaction between the viewer (over WiFi) and broadcaster (who is doing a talent show).

***Message latency (MSL)*:** The text-based messages generated by the viewer are delivered via HTTP. We assume that the viewer gives a gift to the broadcaster to pay for a live song at the time t_1. The WiFi AP, message server and broadcaster receive the message at t_2, t_3 and t_4, respectively. Hence, MSL can be denoted by $t_4 - t_1$. We assume that the broadcaster begins to sing at t_4, just the moment of receiving the gift.

***Video frame latency (VFL) and video frame latency over WiFi (VFLW)*:** The video frames created by the broadcaster

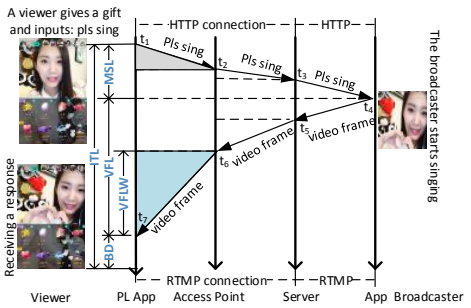


Fig. 2. The breakdown of PL's interactive latency.

are delivered via RTMP [7]. The video contents have been encoded into H.264 format RTMP frames on the broadcaster's PL App and they are then sent to a RTMP server, which receives the first video frame at t_5. The frame is then sent to the client over another RTMP connection established between viewer's PL App and the RTMP server. The WiFi AP along the path receives the first packet of the frame at t_6. After a while, all of the packets of the frame are delivered to the viewer's PL App at t_7. VFL and $VFLW$ can be represented by $t_7 - t_4$ and $t_7 - t_6$, respectively.

***Buffer delay (BD)*:** Finally the viewer watches the broadcaster's show after another BD milliseconds, where BD is the size of playback buffer of the PL's media player.

We focus on $VFLW$ denoted by $t_7 - t_6$, since the last hop frame latency dominates of the ITL (see details in Section III). Due to the space limitations, we omit some details, such as video transcoding on a RTMP server, data copy from TCP buffer to a media player's decoder, initialization of a TCP connection and DNS lookup, *etc.*

III. Motivating Example

As shown in Section II, in China, PL is more monetized in that broadcasting has become a profession. Many broadcasters make a living by actively interacting with viewers (e.g., react to viewers request) and then receiving cash gifts from their viewers. Hence, much lower interactive latency is required to support these activities compared with PL in the United States. However, PL latency in China is little understood because it is infeasible for third parties like us to measure this key metric. The timestamps of frame delivery can be measured at client side in US, because Periscope puts timestamps at the frame creation at the broadcaster's side, while there is no such timestamp for PL in China. Previous studies show that WiFi is the weakest link for *end-to-end* packet latency [5], [6]. To this end, we setup a testbed as shown in Fig. 3 (a) to quantify the contribution of $VFLW$ to ITL using two pairs of laptops and mobile phones, VLC media player [13] and an Android simulator.

To evaluate the impact of last hop WiFi on ITL and to isolate other factors, such as operation system and hardware platform, we connect one laptop to an Open-Wrt AP via WiFi (802.11n, 2.4GHz) and another laptop to an Ethernet network which shares the same backhaul connection. We created a broadcast on one of commercial PLs (*i.e.,* Meipai) using a

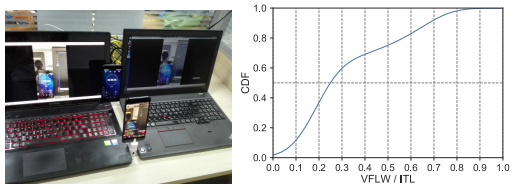


Fig. 3. (a) A pilot testbed to measure interactive latency (*i.e., ITL*) and video frame latency over WiFi (*i.e., VFLW*). (b) The distribution of $\frac{VFLW}{ITL}$, which indicates that $VFLW$ can be the major contributor of ITL.

mobile phone, whose camera points to a running stopwatch. A viewer joins this broadcast with the RTMP URI of the broadcast using VLC media player on each of the two laptops, and the time displayed both at the broadcaster and viewer side are recorded manually. These experiments are repeated 40 times over WiFi and wired networks every 5 minutes between 9:00 a.m to 10:00 a.m, 13:00 to 14:00, and 20:00 to 22:00. We set the playback buffer of VLC as zero and. Hence, BD equals 0 throughout the measurement. This allows us to measure VFL of the two access networks, which can be roughly derived by the difference between viewer time and broadcaster time. Thus, $VFLW$ can be further derived by the difference of VFL over WiFi and wired network. To obtain the MSL, we create a broadcast using Meipai App on a mobile phone and join the broadcast by Meipai App running in the Android simulator, both on WiFi and wired networks. We type a message at the viewer side, and we then record a video of both the viewer's and the broadcast's screen. MSL can be derived by the timestamp difference of sending frame and receiving frame in a recorded video. Finally, we get $ITL = VFL + MSL$ and $VFLW$ of Meipai respectively over WiFi networks.

Fig. 3 (b) plots the distribution of the ratio of $VFLW$ to ITL based on our pilot measurement. It shows that the 75th percentile is 50%, which means that, for about 25% of video frames, the latency at the WiFi hop (*i.e., VFLW*) contributes to more than 50% of the frames interactive latency (*i.e., ITL*). Furthermore, as will be shown in Fig. 6 (c), 80% of the $VFLW$ is the WiFi overhead. These observations motivate us to model and improve the $VFLW$ of PLoW for a good QoE. Although the upstreaming message latency over WiFi (*i.e.,* $t_2 - t_1$ in Fig. 2) can also be impacted by WiFi, the small size of the message makes WiFi's impact much less significant. Therefore, we focus on PL's $VFLW$ in this paper, which can be the major contributor of PL's end to end interactive latency.

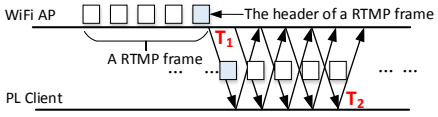


Fig. 4. $T_2 - T_1$ denotes the video frame latency over WiFi, where T_1 and T_2 represents the time when the first packet of a video frame is received by an AP's wired NIC and the last packet of the frame received by the video player, respectively.

IV. Data Collection

In this section, we present and deploy Latency Doctor (LTDr), a system framework which collects, measures, and reduces the PL's $VFLW$. A large dataset is collected by LTDr from three leading PLs in China.

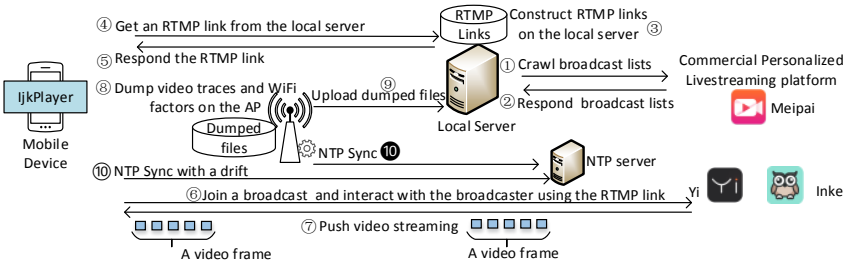


Fig. 5. LTDr's detailed process for measuring PL's frame latency over WiFi (*i.e., VFLW*).

A. Measuring video frame latency over WiFi

As shown in Fig. 4, $VFLW$ can be denoted by $T_2 - T_1$. It equals the time elapsed between the first packet of a video frame arrives at an AP's wired NIC at T_1, and the complete frame is received by a media player at T_2. However, most commercial PL traffic is encrypted in China, making it difficult to obtain T_1 and T_2. To tackle this challenge, we modified the ijkPlayer * [8] to mimic a real viewer who can join and leave a PL broadcast periodically, using the RTMP URLs that are crawled from the unencrypted video APIs of PL platforms. However, identifying the first TCP packet of a RTMP frame from a video trace is challenging because a RTMP header and video data are always encapsulated in the same packet. To tackle this challenge, our proposed LTDr compares and searches the binary data in a video trace according to RTMP specification [7] to pinpoint this frame boundary.

Measuring T_1: To obtain real video traces on our testbed, we continuously crawl unencrypted broadcast list pages of Meipai, Inke and Yi (Step ① and ② in Fig. 5), and we then construct the RTMP links based on the crawled URLs (Step ③ in Fig. 5). We customized the ijkPlayer [8] and made it run on a mobile phone to periodically access RTMP links to mimic the user's viewing actions, such as joining and leaving every 5 minutes (Step ④ to ⑦ in Fig. 5). The video traces are captured by *tcpdump* on the connected AP's wired NIC and then they are uploaded to a server periodically (Step ⑧ and ⑨ in Fig. 5). During offline analysis, the server parses video traces and identifies the first TCP packet of each RTMP to obtain a frame number and its receiving time T_1 on AP's wired NIC, referring to the RTMP specifications. The system time of an AP is synchronized with an NTP server (Step ⑩ in Fig. 5).

TABLE I
DATASET COLLECTED FROM MEIPAI, INKE AND YI

Dataset	WiFi Environment	Viewed broadcasts	Video frame
I1	Real environment	10000	9.6M
I2	Injected interference	2000	4.3M

Measuring T_2: The customized ijkPlayer logs each RTMP frame's number and its receiving time T_2 when a completed video frame is constructed in the media player from the TCP buffer. Although the NTP client on an Android phone can record the phone's clock drift, there is no Android interface to actually adjust the phone's clock conveniently. Instead, we send the recorded drift to the data collection server which adjusts T_2 timestamps accordingly (Step ⑩ in Fig. 5).

*The ijkPlayer is an open sourced media player that is designed for PL, and is used by bilibili, a very famous video streaming platform in China.

B. Deployment and dataset

We deploy LTDr in the Tsinghua campus network, consisting of 12 mobile phones and 12 Open-Wrt based NETGEAR4300 APs (802.11n, 2.4GHz) located in four labs to capture video traces and WiFi factors, two servers to crawl RTMP URIs and computing $VFLW$. We focus on 2.4GHz WiFi in this paper because it is more crowded than 5GHz band, and it is still much more popular than 5GHz in our campus. However, the methodologies and predictive models verified under 2.4GHz band could also be easily extended to a crowded 5GHz WiFi network. The WiFi factors (See details in [5]), including Channel Utilization (CU), Interference Utilization (IU), Queue Length (QL), Retry Ratio (RR), Transmitting Throughput (T_{tx}), Receiving Throughput (T_{rx}), RSSI, Receiving Physical Rate (RPR) and Transmitting Physical Rate (TPR), are captured using the Linux commands on each AP every 100ms. In total, 12 thousand broadcasts have been viewed by the customized ijkPlayer from 12th December 2016 to 10th February 2017 and the corresponding video traces are reconstructed into millions of video frames with $VFLW$. Details of the Dataset I1 are shown in Table I.

C. Distributions

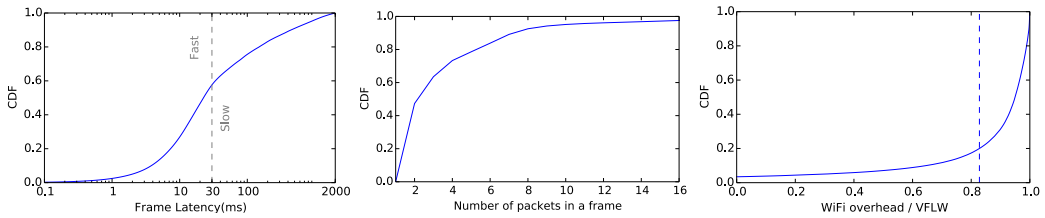
Fig. 6 (a) plots the distribution of $VFLW$. It shows that 40% of $VFLW$ is larger than 30ms. A previous study has shown that [12] the *end-to-end* latency of interactive online gaming should be less than 60ms to ensure a good user experience, which, although from a different domain, helps calibrate how large a 30ms $VFLW$ is. Fig. 6 (b) depicts the distribution of packet count in video frames of PL. It shows that about 40% of the frames consist of more than three packets. Fig. 6 (c) presents the distribution of WiFi overhead of a frame (including channel backing off time, delay caused by re-transmission, *etc*). It shows that WiFi overhead can contribute to more than 80% of $VFLW$.

In summary, the findings in Fig. 6 highlight the importance of understanding WiFi's radio impact on PL's $VFLW$. We can also conclude that the PL's $VFLW$, as shown in Fig. 2 and Fig. 4, is quite different from WiFi's per-packet latency, which can be simply measured by the *ping* command on a mobile client.

V. MODELING

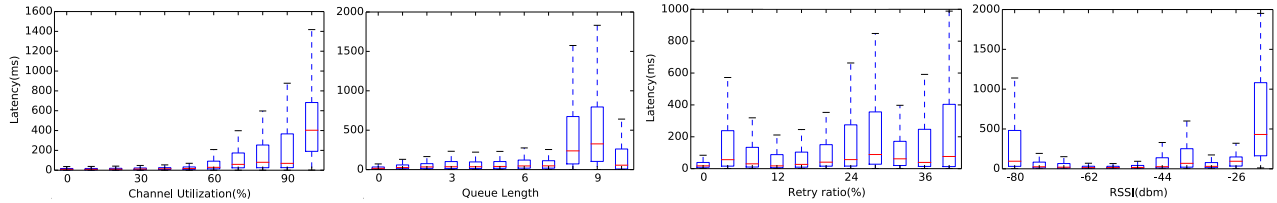
A. Enriching the parameter space

Our dataset collected from the campus testbed can hardly cover all of WiFi parameter space. To mitigate this limitation, we deploy a traffic generator and introduce background traffic and wireless interference for a broad range of parameter



(a) (b) (c)

Fig. 6. (a) Distribution of the video frame latency over the WiFi hop; (b) Distribution of packet count per frame; (c) Distribution of WiFi overhead in $VFLW$.



(a) (b) (c) (d)

Fig. 7. Relationship between WiFi factors (x-axis) and PL's $VFLW$ (y-axis).

settings. As will be discussed in Section V-D, this work can also help to build a more accurate and robust machine learning model because a training set is expected to cover most of the parameter settings [14]. To this end, we additionally deploy a WiFi traffic generator and introduce background traffic and wireless interference. The generator contains 30 NETGEAR4300 WiFi routers, which are divided into three groups, each containing 10 routers. We set wireless channel number as 1, 6 and 11 for each group, respectively. In each group, five routers are set as wireless clients and other five are configured as servers. Each client associates with an individual server. Then, a client sends UDP requests to the corresponding server using the *iperf* tool, with rates ranging from 1Mbps, 2Mbps,...,16Mbps. We then run LTDr and deploy the traffic generators from 10th February to 10th March 2017. Table I depicts Dataset I2 collected by the modified ijkPlayer.

B. Single-dimension analysis

Fig. 7 depicts the relationships between WiFi factors and PL's $VFLW$ based on Dataset I1. We summarize the main take-aways as follows.

1. Heavy workloads lead to higher *VFLW*: As shown in Fig. 7 (a), increasing Channel Utilization (CU) will significantly increase the $VFLW$. These results suggest that CU should be under a threshold to reduce $VFLW$ at a certain level. We will discuss these values in Section V-E.

2. Queue length and retry ratio correlate well with *VFLW*: As shown in Fig. 7 (b) and Fig. 7 (c), longer queue length causes higher $VFLW$ since increasing the length will introduce more waiting time on an AP. Higher retry ratio results in higher *VFLW*, which can be caused by the interference, fading and noise problems.

3. RSSI does not correlate well with *VFLW*: As shown in Fig. 7 (d), $VFLW$ fluctuates when RSSI increases. Surprisingly, there is a latency peak at -38dB RSSI. Our investigation shows that the CU is higher at this part of the dataset. Such result shows the ineffectiveness of single-dimension analysis in a complex wireless environment.

TABLE II
KENDALL CORRELATION SCORE(KCS) AND RELATIVE INFORMATION GAIN(RIG).

WiFi	QL	CU	RPR	IU	RR	RSSI	T_{rx}	TPR	T_{tx}
KCS	0.44	0.39	-0.37	0.37	0.15	0.14	0.14	0.01	0.05
RIG	0.06	0.06	0.05	0.04	0.003	0.02	00	-0.06	0.02

To quantify the relationship between WiFi factors and $VFLW$, we consider the Kendall Score (KCS) and Relative Information Gain (RIG) that are used in [15]. Table II shows the results of these measurements, which are listed in a descending order of KCS, with outstanding values highlighted in blue.

In summary, we have studied the complex correlations between each WiFi factor and $VFLW$ based on a dataset collected from three leading PL platforms in China. However, such single-dimension analysis is not enough and, hence, the quantifications on multiple-dimensions are expected to build a simple yet accurate predictive model. Therefore, in the following parts, we use machine learning methodologies to quantify on multiple-dimensions and build a simple yet accurate predictive model

C. Modeling overview

As previously discussed, accurately predicting $VFLW$ is challenging due to the complex relationships between highly dynamic WiFi factors and $VFLW$, and these relationships are non-linear or even non-monotonic. Furthermore, there exist interdependencies between various wireless network parameters [5]. To tackle these challenges, we use different supervised learning algorithms to evaluate the prediction performance in terms of recall and precision, including Random Forest, Linear Supporting Vector Machine (L-SVM), Decision Tree and Logistic Regression. We detail the modeling in the following parts of this section.

D. Selecting machine learning algorithm

Labeling: Previous study has shown [12] that a user perceptive experience of interactive gaming is good within 60ms latency,

and is bad larger than 100ms, and 150ms or more will significantly degrade user engagement by 75%. Due to the lack of such a reference for PL, we borrow these numbers for the labeling of PL's data. Consequently, as shown in Fig. 6(a), we label the $VFLW$ of the dataset into binary categories for practical usage, including *FAST* and *SLOW* with a threshold configured as $30ms$, which is half of the expected latency for a good interactive experience.

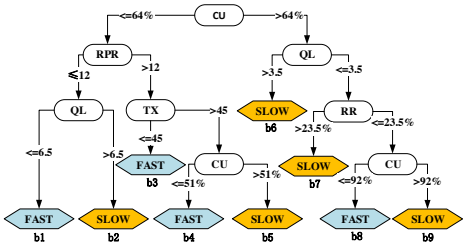


Fig. 8. Pruned Decision Tree for classifying $VFLW$ into FAST and SLOW.

Data preprocessing: We conduct Z-normalization and one-hot encoding for the large volume of the dataset with more than 13.9 million items. The dataset is encoded into *libsvm* format. We also clean the data with some very abnormal values, such as the $VFLW > 2s$ caused by network interruptions, abnormal NTP time offset, *etc*.

Evaluating machine learning models: We run Random Forest, L-SVM, Decision Tree and Logistic Regression algorithms, respectively, on a cluster using Apache Spark's scalable machine learning library (Spark MLlib) [16]. We evaluate these models using 10-fold cross validation on the dataset. The precision and recall curve in Fig. 9 shows that, among the four algorithms, both Random Forest and Decision Tree algorithm perform best over the dataset. This shows that both of these metrics can reach 0.81 simultaneously. Thus we select Decision Tree considering its advantage of easy interpretation. We plot the pruned Decision Tree in Fig. 8. Using this visualized classification.

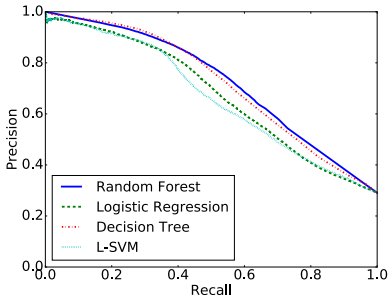


Fig. 9. Performance comparisons of different machine learning algorithms.

E. Observations

The Decision Tree model shown in Fig. 8 presents some interesting insights as follows.

1) CU should be lower than 64% to achieve good *VFLW*: Most of *SLOW* results appear in the right branch of the Decision Tree where the CU is larger than 64%. Hence, the CU should not exceed this threshold for a good interactive experience (see branches *b*6, *b*7 and *b*9).

2) QL larger than 3.5 may lead to slow *VFLW* (branch *b*6)**:** We observe that a larger QL (*i.e.,* > 3.5) with a higher CU (*i.e.,* $> 64\%$) leads to *SLOW* results. This is reasonable because the input traffic from media servers to an AP is always kept constant while the output speed is significantly impacted by the workload of the wireless channel. Hence, a higher CU leads to a larger QL and introduces additional delay.

3) Smaller RPR with a smaller CU may also result in slow *VFLW*: The Decision Tree model suggests a *SLOW* result when CU $< 64\%$, RPR $< 12Mbps$ and QL > 6.5 (branch *b*2), while a larger RPR (*i.e.*, $> 12Mbps$) will lead to *Fast* results. We explain this as follows. RPR represents the physical sending rate of a mobile station and it is controlled by the rate adaptation algorithms [17] of IEEE 802.11-based wireless networks. The algorithms estimate the channel quality and adjust the transmission rate (*i.e.*, RPR) accordingly. A smaller RPR (*i.e.*, $<= 12Mbps$) indicates a lower network quality and, hence, may lead to a higher $VFLW$.

In summary, our simple yet accurate Decision Tree model, as shown in Fig. 8, can be used to guide the optimization of PL's $VFLW$.

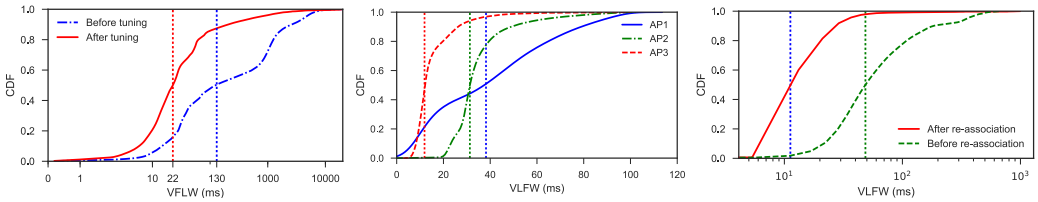
VI. Improvement and Evaluation

In this section, we highlight two real cases to demonstrate our learning-based optimization to help WiFi owners or users to understand, troubleshoot, and mitigate the WiFi hop $VFLW$ in the wild.

A. Reducing VFLW in a residential wireless network

Generally, in a residential wireless network, there is a single access point available for users. For a specific AP, we need to map its specific packet's WiFi factors onto the tree in Fig. 8, and then find its specific path from the root to the leaf. In the following case, we take two steps to improve PL's $VFLW$ in a residential wireless network based on the Decision Tree model.

Classification: In our university, multiple WiFi APs are deployed on each floor of the student dormitories, and a student always connects his laptop or smartphone to one of the APs to surf the Internet. However, some students complain that the ITL of Inke is too large for smooth interactions with broadcasters in their dormitory (*e.g.,* Room 614) from 19:00 to 22:00. They have no idea about what causes the large latency of PL. Hence, we use the Decision Tree model to diagnose the network and prioritize the WiFi factors. In the diagnosis, the time and place are important because the students need to know when and where high WiFi latency is likely to occur so that they can take further actions such as troubleshooting. We collect WiFi factors on the AP for two days from 19:00 to 22:00 and classify them using the model in Fig. 8. The percentile in the Table III shows the fraction of video frames that falls into each specific branch in Fig. 8. In this case, we find that most (*i.e.,* 61%) of the results fall into the *SLOW* branch *b*7, which indicates a combination of $CU > 64\%, QL <= 3.5$, and $RR > 23.5\%$.



(a) (b) (c)

Fig. 10. (a) $VFLW$'s distribution before and after optimization; (b)$VFLW$'s distribution of three APs; (c) $VFLW$'s distribution before and after re-association.

TABLE III
PERCENTAGE OF PACKETS THAT FALL INTO EACH BRANCH

Branch	b1	b2	b3	b4	b5	b6	b7	b8	b9
Before	03	03	0	01	19	02	**61**	12	0
After	09	0	0	01	19	02	00	**69**	0

Diagnosis: Based on the classification results, we are able to infer the root cause of *SLOW VFLW* on the targeting AP. In the branch $b7$, $CU > 64\%$ at the root node indicates heavy workload of the channel, hence, this factor should be firstly considered as a potential cause of *SLOW VFLW*. *QL* represents the length of packets queued in an AP's sending buffer and a large value may increase the latency. However, we ignore this feature because it is smaller than 3.5 (*i.e.,* $<= 3.5$). A large *RR* can be caused by many problems, such as heavy load, hidden terminal, low SNR, and local contention problem [6]. We jointly analyze $CU > 64\%$ and $RR > 23.5\%$ and guess that a larger *RR* may be caused by a heavy load or local contention problem, which increases collision probability and the backoff waiting time of a wireless channel by a large number of concurrent senders, and which also decreases the achievable channel utilization. Hence, we try to switch the channel of the AP in Room 614 from number 1 to 11 and try to find if the playback of PL becomes more smooth. To get the ground truth of the latency, we parse the video traffic and get the classifications as shown in the second row of Table III. Now most (*i.e.,* 69%) of the $VFLW$ falls into the branch $b8$, which means $64\% < CU < 92\%, RR <= 23.5\%$, and $QL <= 3.5$. Hence, *RR* is the critical feature that causes the *SLOW VFLW* in this case.

Improvement: The distributions of $VFLW$ before and after changing the targeting AP's channel are depicted in Fig. 10 (a). This shows that the median value has decreased from 130ms to 22ms after optimization. Hence, our predictive model can help to prioritize the WiFi factors to improve the PL's latency in a residential wireless network. Such model can be extended to facilitate the performance optimization of other latency sensitive applications, such as cloud gaming, video conferencing, and so on.

B. Reducing the VFLW in an EWLAN

In an EWLAN, multiple APs with different channel conditions are available for mobile users. Hence, it is feasible to associate a client to an AP with better performance to improve the PL's $VFLW$. The default RSSI-based AP selection scheme using signal strength cannot efficiently improve the PL's latency due to the complex relationships between WiFi factors and $VFLW$. In this scenario, our proposed machine learning model can help a PL application to make a right decision to achieve low $VFLW$ by re-associating to the AP with the best performance.

To this end, we design and implement RLive, a practical system which leverages the proposed machine learning model to mitigate the negative impact of WiFi factors on $VFLW$. RLive consists of two main parts: the RLive application on a user device and RLive controller on a server. The working process of Rlive is shown in Fig. 11 and the details are described as follows. Here, we assume the APs in EWLAN can collect WiFi factors (using SNMP from commercial vendors or using methods listed in [5] on Open-Wrt-based APs).

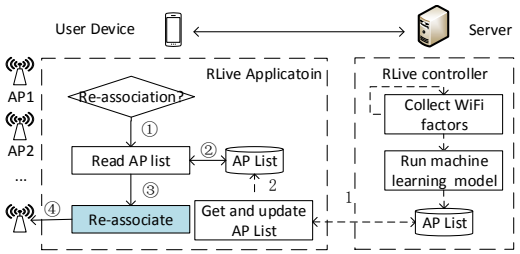


Fig. 11. Working process of Rlive.

RLive controller: This controller aims to guide a mobile client to select an AP with the lowest $VFLW$. Hence, it periodically predicts the $VFLW$ for each AP, using the Decision Tree model and WiFi factors collected from APs (via SNMP in EWLAN), and then updates the performance results in an AP list.

RLive application: This application aims to help a mobile phone associate to a candidate AP to reduce the $VFLW$, without making any changes in the operating system kernel or drivers. As shown in Fig. 11, the modified ijkPlayer triggers a re-association event when a video quality degradation occurs, such as three buffering events in 1 minute, where the threshold can be set by user's viewing preferences. The RLive application reads the AP list when receiving an event, where the list is periodically updated from the RLive controller. Then, the RLive application provides the user with an option to associate or not. If the user clicks the YES button, then the RLive application re-associates to the AP with the best performance.

Deployment: In the department building, some of the students complain that ITL of Inke is too large and they suffer too many playback bufferings. In this case, we demonstrate how RLive improves PL's latency using our machine learning model in an EWLAN, and such test can represent scenarios that a user is capable of switching among different APs on an EWLAN where multiple APs are available. Our testbed consists of three mobile phones (*i.e.,* MP1, MP2 and MP3),

on which we have installed a modified ijkPlayer and RLive application, three Open-Wrt APs (*i.e.,* AP1, AP2 and AP3) that are configured to channel 1, 6 and 11, respectively, and a remote server that acts as RLive controller. Mobile clients MP1, MP2 and MP3 connect to AP1, AP2 and AP3, respectively. During the experiment, the three clients join broadcasts of Inke using the modified ijkPlayer. We found that MP1 suffers many playout bufferings during a viewing. It is noteworthy that playback buffering has a positive correlation with $VFLW$ and can be detected by a media player.

Improvement: To improve the $VFLW$, the RLive application on MP1 sends an alert button to connect AP3. To validate our predictive model, we capture the video traces on each AP to get $VFLW$ as ground truth (see details in Section IV) before a re-association. The distributions of $VFLW$ over three APs are shown in Fig. 10 (b), and we can observe that the median value of $VFLW$ over AP1 is the largest, while the value over AP3 is the smallest. We found that overall ITL over MP1 is reduced after clicking the YES button and re-associating to AP3. Fig. 10 (c) depicts the performance comparison before and after MP1's re-association, and it shows that the median $VFLW$ is significantly decreased from 50ms to 12ms.

In summary, our simple yet accurate Decision Tree model, which we built in Section V, can significantly improve the PL's $VFLW$ in the wild and, hence, reduces overall *end-to-end* interactive latency ITL. We highlight that the AP's re-association management in the second case is based on the predictive model running on a server, without making any changes in operating system kernel or drivers in mobile devices. Furthermore, RLive works in an online mode on the mobile client side and, hence, differs from commodity WiFi routers, such as Cisco Prime and Aruba, which always dynamically select a spare channel for an AP during startup via a centralized network controller.

VII. Related work

Measurement and optimization of PL: Previous studies [1], [2] have conducted detailed analysis of mobile PL based on two leading applications: Periscope and Meerkat. We differ from these works in the video frame latency measures and modeling over last hop WiFi (*i.e., $VFLW$*). Furthermore, our dataset is collected from three leading PL platforms in China, whose characteristics are quite different from those of Periscope and Meerkat in [1], [2].

Studies of other interactive applications: A previous [18] study has used multiple wireless links to improve VoIP streaming quality over WiFi. Outatime is proposed in [12] to reduce online gaming latency over mobile network. The authors of [19] measured mobile video call applications over WiFi and cellular networks. However, we improve the PL's $VFLW$ with encrypted video traffic by model-based optimization, without requiring any changes on a WiFi AP.

Characterizing WiFi networks: Past works have extensively studied WiFi performance, such as last hop latency measured by TCP three-ways handshakes over a campus WiFi [5] and *ping* based instrument over an EWLAN [6]. In this paper, we model the relationships between fine-grained WiFi factors and PL's $VFLW$ using the dataset collected by a media player, in contrast to script-based packet level measures.

VIII. Conclusion

We present LTDr, a system framework which collects, measures, and further reduces the PL's $VFLW$ to improve the PL's QoE. Our work is motivated by a controlled experiment which shows that $VFLW$ is the major contributor of the PL's interactive latency. To this end, we build a predictive model based on a real dataset, and verify its effectiveness by two real cases. Such a model can help WiFi protocol designers, WiFi network owners and designers, and PL applications to monitor or improve PL's latency. LTDr is general enough and can be extended for modeling and improving the latency of other interactive video streaming services over WiFi, such as video conferencing and mobile cloud gaming. In the future, we plan to work with AP manufacturers and commercial PL platforms to develop a scaled solution with further generalized models that can reduce interactive latency.

IX. Acknowledgement

The work was supported by National Natural Science Foundation of China (NSFC) under grant No. 61472214, No. 61472210, and No. 61671081, and the Beijing Natural Science Foundation under Grant No. 4172042.

References

[1] B. Wang *et al.*, "Anatomy of a personalized livestreaming system," in *IMC '16*. New York, NY, USA: ACM, 2016, pp. 485–498.
[2] M. Siekkinen, E. Masala, and T. Kämäräinen, "A first look at quality of mobile live streaming experience: The case of periscope," in *IMC '16*. New York, NY, USA: ACM, 2016, pp. 477–483.
[3] "Periscope," https://www.periscope.tv/, [Online].
[4] "Cisco VNI Global Mobile Data Traffic Forecast, 2016-2021," http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, [Online].
[5] C. Pei, D. Pei *et al.*, "Wifi can be the weakest link of round trip network latency in the wild," in *INFOCOM 2016*, April 2016, pp. 1–9.
[6] K. Sui, D. Pei *et al.*, "Characterizing and improving wifi latency in large-scale operational networks," in *MobiSys '16*. New York, NY, USA: ACM, 2016, pp. 347–360.
[7] "Real-time messaging protocol (rtmp) specification," http://www.adobe.com/devnet/rtmp.html, [Online].
[8] "ijkplayer," https://github.com/Bilibili/ijkplayer, [Online].
[9] "Inke," http://www.inke.cn/, [Online].
[10] "Meipai," http://www.meipai.com/, [Online].
[11] "Yi," http://www.yizhibo.com/, [Online].
[12] K. Lee, Chu *et al.*, "Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming," in *MobiSys '15*. New York, NY, USA: ACM, 2015, pp. 151–165.
[13] "Vlc media player," http://www.videolan.org/vlc/, [Online].
[14] X. S. Wang *et al.*, "How speedy is spdy?" in *NSDI '14*. Seattle, WA: USENIX Association, 2014, pp. 387–399.
[15] F. Dobrian, V. Sekar *et al.*, "Understanding the impact of video quality on user engagement," pp. 362–373, 2011.
[16] "Spark mllib," http://spark.apache.org/mllib/, [Online].
[17] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan, "Robust rate adaptation for 802.11 wireless networks," in *MobiCom '06*. New York, NY, USA: ACM, 2006, pp. 146–157.
[18] R. Kateja *et al.*, "Diversifi: Robust multi-link interactive streaming," in *CoNEXT '15*. New York, NY, USA: ACM, 2015, pp. 35:1–35:13.
[19] C. Yu, Y. Xu, B. Liu, and Y. Liu, "can you see me now? a measurement study of mobile video calls," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 1456–1464.