FedLive: A Federated Transmission Framework for Panoramic Livecast with Reinforced Variational Inference

Xingyan Chen, Mu Wang, Changqiao Xu, *Senior Member, IEEE,* Yu Zhao†,
Ke Jiang, Yang Shujie, Qi Li, Lujie Zhong, Gabriel-Miro Muntean, *Fellow, IEEE,*

***Abstract*—Providing premium panoramic livecast services to worldwide viewers considering their ultra-high data rate and delay-sensitivity is a significant challenge in the current network delivery environment. Therefore, it is important to design an efficient way of improving viewer quality of experience while conserving bandwidth resources. In this context, this paper introduces a novel cost-efficient federated transmission framework called *FedLive* and a set of algorithms to support it. First a gradient-based clustering method is proposed to group the geo-distributed viewers with similar viewing behavior into content delivery alliances by exploiting the geometric properties of the gradient loss. Next, a Reinforced Variational Inference (RVI) structure-based approach is proposed to assist with the collaborative training of the viewer field of view (FoV) prediction model while also accelerating the tile delivery process. A novel prediction-based asynchronous delivery algorithm is designed in which both the high accuracy FoV prediction and efficient live 360° video transmission are achieved in a decentralized manner. FedLive was implemented for testing and an open source code is made available. Finally, the proposed solution was evaluated against a benchmark and three alternative state-of-the-art solutions using a real-world dataset. The experimental results show that our approach provides the highest prediction accuracy, better service performance, and saves bandwidth when compared with the other solutions.**

I. Introduction

Supported by the latest evolution of wireless communication and hardware solutions, Panoramic Livecast Services (PLS) is a new type of immersive and interactive entertainment which is gaining increasing popularity. Nowadays, users can access easily rich live 360° video content from commercial content providers, such as YouTube VR[1], Google VR[2], NYT News[3], via various types of tethered and wireless Head-Mounted Displays (HMD). The rapid development of HMD devices offers the public feasible opportunities to interact with the Metaverse, seen as the next-generation Internet which supports the vision of future human life and work [1]. However, supporting the ultimate panoramic video viewing experience is very challenging in terms of current livecast delivery. For instance, an ultra high definition or a 12K immersive video experience requires more than 680 Mbps bandwidth [2], and has stringent latency requirements (i.e. reaction within 20ms and at least 60Hz refresh rate) to avoid motion sickness [3]. Providing premium PLS consumes almost ten times the amount of data compared with the current worldwide average bandwidth [4]. In addition, multi-resolution panoramic livecast deployment requires, apart from bandwidth, intensive computing support as well, in order to match different configurations of networks and end devices. Solutions are sought to reduce this dual communication and computation resource pressure.

Since a viewer only watches a small portion of the sphere video at any given time, viewport prediction is a promising technology for PLS [5]. Predicting viewers' Field of View (FoV) allows the PLS system to stream high-resolution FoV and low-resolution background to the viewers, saving bandwidth resources and reducing the motion-to-photon delay [6]–[9]. However, the performance of these solutions mainly hinges on the FoV prediction accuracy, since inaccuracy leads to mismatch tiles fetching, which further determines waste of bandwidth resources. To improve the accuracy, Hou *et al.* in [6] designed a deep learning-based viewpoint prediction method to facilitate adaptive 360° video streaming in mobile networks. Fan *et al.* [7] jointly considered head movement patterns and saliency features of the viewers in their prediction model design, and Nguyen *et al.* [8] employed the inter-dependency between the head motion and saliency features in their solution. Wu *et al.* in [10] proposed a preference-aware viewport prediction network by extracting the saliency features of 360-degree video. Other prediction paradigms such

X. Chen, Y. Zhao and Q. Li are with Financial Intelligence and Financial Engineering Key Laboratory of Sichuan Province, Institute of Digital Economy and Interdisciplinary Science Innovation, Southwestern University of Finance and Economics, China. E-mail: {xychen, zhaoyu, Liq_t}@swufe.edu.cn.

M. Wang, C. Xu, K. Jiang and S. Yang is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, P. R. China. E-mail: cqxu@bupt.edu.cn, muwang@tsinghua.edu.cn, {jke, sjyang}@bupt.edu.cn.

L. Zhong is with Information Engineering College, Capital Normal University, Beijing 100048, P. R. China. E-mail: zhonglj@cnu.edu.cn.

G.-M. Muntean is with School of Electronic Engineering, Dublin City University, Glasnevin, Dublin 9, Ireland, E-mail: gabriel.muntean@dcu.ie

†The corresponding author is Yu Zhao.

[1]https://vr.youtube.com

[2]https://arvr.google.com/vr

[3]http://www.nytimes.com/marketing/nytvr

as involving viewer observation location [9], user individual behaviours [11], and other evaluation metrics for 360 video streaming [12] are also considered in alternative solutions. Although the aforementioned solutions provided satisfactory prediction accuracy, they all follow a *train-and-then-predict* approach, which requires information about the whole video and sufficient number of user viewing records and introduces a large model training delay. Therefore this approach cannot be directly applied to livecast services [13], [14].

To address this issue, new online solutions are proposed in the literature [14]–[19]. For example, Feng *et al.* [14] presented a central online FoV prediction solution for live virtual reality (VR) streaming called LiveDeep. The solution introduced three functions including an online FoV prediction model, a lifelong model update, and a real-time FoV inference mechanism. In the quest to improve the performance, reinforcement learning (RL)-based solutions have also been proposed [15]–[20]. Feng *et al.* introduced an object semantics-based FoV prediction approach that employs a 4-tuple (state, action, policy, reward) representation of live mobile VR video streaming in a RL problem formulation [15]. Then, the authors designed a RL-based algorithm to predict user region of interest (ROI) and control tile-based content delivery. Jiang *et al.* [16] proposed a novel deep RL framework to optimize online video streaming in terms of viewport prediction, prefetch scheduling, and rate adaptation. Two centralized multi-model deep RL frameworks are provided by Pang *et al.* [17] and Zhang *et al.* [18] to capture interactive features of live 360° video and optimize QoE in terms of multiple metrics, respectively. However, supporting premium PLS requires employment of centralized solutions to offer online FoV prediction for each individual, which is extremely expensive. Scalability issues arise as the number of viewers grows. Therefore, Ban *et al.* [20] designed a Multi-Agent deep RL-based framework to improve training efficiency of their prediction model by offloading the training tasks to large number of viewers. Some researches [21], [22] suggested leveraging the similarity of user viewing behaviour to overcome the scalability challenge. Xie *et al.* [22] conducted an analysis of the viewer behaviour similarity and proposed a novel cross-user learning framework to perform online training based on a unified prediction model for viewers with similar viewing behaviours.

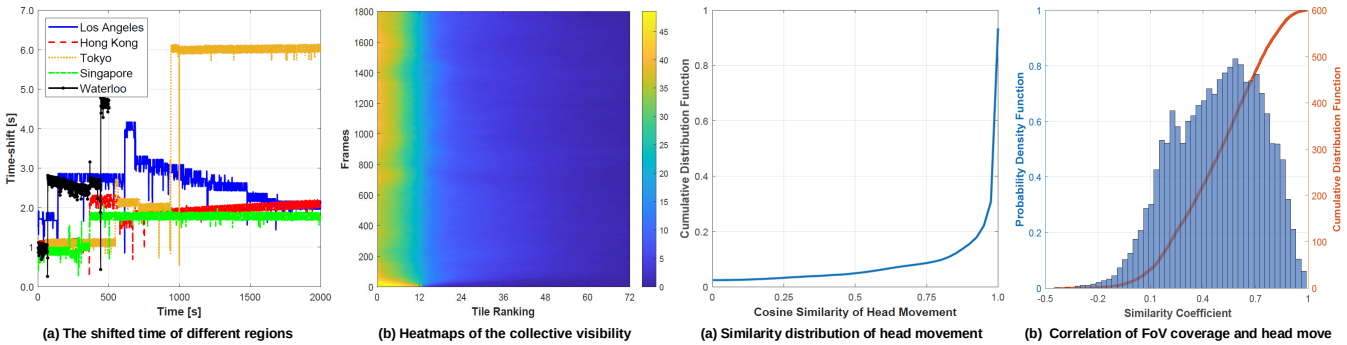
However, the above-discussed solutions do not consider the fact that the live panoramic videos are delivered over an asynchronous networked system in which the viewers are geographically dispersed. Viewers in different geographical locations have diverse stream latencies and time lags between different viewers exist. Ignoring the time lag may make the similarity analysis and the collaborative model training difficult in terms of deployment since the viewers are often in different playback states and have time-varying personalized actions. We believe that PLS should take into account of the asynchronous nature of content distribution and find a new research avenue for prediction model training and data delivery. Yet, most of the current collaborative 360° video transmission solutions are synchronous [23]–[28]. For instance, Mao *et al.* [23] presented a motion-prediction-based multicast approach to transmit cooperatively 360° videos to multiple viewers. Chen *et al.* [24] considered a system that provides multi-users panoramic video streaming services through a single wireless access point and designed a wireless scheduling algorithm to deliver tiles for multiple users based on prediction results. In addition, some innovative video streaming technologies such as scalable video coding [25], [26], shared coded picture [27], caching [28] and super-resolution [29], [30] were also designed to enhance collaborative transmission in synchronous manner. As a synchronous design reduces the flexibility of live 360° video delivery in networked PLS systems, it motivates us to design a new collaborative data transmission framework for PLS that is online, asynchronous, and bandwidth-saving.

This paper proposes a novel asynchronous cost-efficient federated transmission framework called FedLive for PLS. FedLive includes two major contributions: a Gradient-based Viewer Clustering (GVC) method in order to improve communication efficiency for both model training and data delivery, and a Reinforced Variational Inference (RVI) structure-based asynchronous delivery approach whose application is a Prediction-based Asynchronous Delivery (PAD) algorithm for collaborative FoV prediction model training and tile-based data transmission. GVC groups the viewers into networked alliances based on the similarity of their viewport and head-motion actions. The similarity of behavior can be expressed as the cosine similarity of the gradient loss between different viewers based on clustered federated learning [31]. RVI targets FoV prediction, which is a typical probability inference problem since each of video tiles has a certain probability of being watched by the viewers. It uses probabilistic inference algorithms to optimize model training, and follows the footsteps of the famous computer scientists David Silver [32] and Sergey Levine [33], by expanding the design space of optimal control by bridging the gap between variational inference and RL. The RL framework offers an interactive evolutionary learning design for probabilistic inference in problem-solving and brings the potential for devising an online asynchronous algorithm with inference technology. In this context, reinforced variational inference (RVI) is a novel structure-based joint prediction and delivery approach which optimizes the process of distributed online learning and tile-based delivery by formulating the joint optimization problem in the RVI framework space. To the best of our knowledge, this work is the first attempt to apply clustered federated learning and RVI on PLS. GVC and RVI are deployed by the proposed FedLive framework, which is evaluated through both numerical simulations with the benchmark and prototype system experiments. The experimental results show that FedLive outperforms several state-of-the-art solutions [10], [14], [15] in terms of prediction accuracy, bandwidth consumption and delay reduction.

II. Problem Statement and Motivation

In this section, we first introduce briefly PLS. Next, we analyze the time-shift phenomenon and behavior similarity, then explain how these factors motivate us to design FedLive.

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2023.3237325



(a) The shifted time of different regions (b) Heatmaps of the collective visibility (a) Similarity distribution of head movement (b) Correlation of FoV coverage and head move

Fig. 1. Comprehensive analysis for the collecting data and the publicity available dataset [34].

A. Overview of Panoramic Livecast Services (PLS)

A scenario in which content providers, like Twitch, provide various types of live panoramic video content such as sports, online games, education, etc. is considered. Worldwide viewers can access PLS and enjoy live 360-degree videos using HMDs. The panoramic video is generally captured in real-time by an omnidirectional camera and contains every direction of view surrounding the camera. As they are more than just display terminals, HMDs also provide head- and eye-tracking functions enabling sensing of user viewing orientation and viewpoint in real-time. With this information, HMDs identify the viewer region of interest (ROI) and render the corresponding portion of frames on their screen. Hence, PLS inevitably needs to provide viewers with the ability to explore freely the world of the panoramic video to a certain extent. At the same time, this freedom service paradigm poses a challenge in terms of efficient data delivery in the networked PLS system. Driven by the viewers' differentiated preference and random behavior, PLS is required to provide each individual with personalized 360° live streaming.

Video tiling is one of the most common solutions to realize personalized PLS streaming by cropping the 360° video frames into multiple multi-resolution tiles under equirectangular projection and serving the viewers with specific tiles [35]. Since the regions the viewer is interested in are limited to about 11.6%-26.7% of the whole panoramic image [36], the PLS providers can also conserve bandwidth by serving the viewers with low-definition tiles for the background and high-resolution tiles within the region of interest. In addition, video tiling-based schemes also enable the PLS system to reuse tiles among users with similar viewing behavior. However, there is a need to design such a data transmission solution which uses a *predict-and-then-deliver* approach to effectively identify and distribute the multiple video tiles generated on the fly based on user viewing characteristics in order to achieve high-quality PLS. Providing premium PLS based on the tile-based streaming requires support from two key technologies: 1) a time-sensitive and high-accurate FoV prediction approach that can forecast the demand tiles of each viewer in real-time; 2) a tile-based efficient data transmission that can flexibly adjust the delivery path of tiles to accommodate the time-varying network conditions and viewing behaviour of global viewers.

B. Motivation

The previous analysis indicates that the solution design must be performed based on the knowledge of the changing pattern of user viewing behavior and network conditions. In this context, we collected latency data between the content provider and its viewers located in different regions. Then, a trace-driven study was conducted to profile the users and inform the *FedLive* design based on the collected data and dataset [34].

Time-shift phenomenon. The PLS system is responsible for collecting the panoramic video content and for distributing the content to regional data centers such as content delivery network (CDN) servers. Once the content arrives at the regional server, it needs to be converted into the multi-resolution tiles and then waits for viewer requests. From collection to finally reaching the viewers, the content lifetime differs for each viewer since it is susceptible to many factors including geographical location, heterogeneous network conditions, and workload of the servers. In other words, there is a time lag between different viewers requesting and receiving the same frame of the video. We call this aspect *time-shift phenomenon* and the time lag between different viewers as *shifted time*. To verify the time-shift phenomenon, we crawled the lifetime variation of the content from a broadcaster to its viewers in Twitch.tv and the results are shown in fig. 1 (a). The viewers are equipped with different devices and are from different geographical regions including North Asia (Tokyo), North America (Los Angeles, Waterloo), and Asia-Pacific (Hongkong, Singapore). This results indicate that the *time-shift phenomenon* is common and that the *shift time* can be even up to 4 seconds or more due to the geo-distribution. It is estimated that the *time-shift phenomenon* will be more serious for bandwidth-intensive PLS.

Similar behaviour. The analysis of similarity in viewer viewing behaviors is based on a public dataset [34], which contains the head movement trajectories of 48 users for 18 different videos. The corresponding viewpoint and FoV of each viewer under equirectangular projection can be generated by the rules provided in [37]. Starting from these time-series records, we analyze the FoV coverage rate between different viewers, the similarity of the viewpoint movement and the correlation coefficient of the above two factors. Fig. 1 (b) shows the tile-based heatmap of the average collective

visibility for all videos. In Fig. 1 (b), X-axis represents the rank order of frame tiles, and Y-axis represents the video frames' time slot. The more popular the tile is, the smaller the number of the rank index. Thus the "tile 0" represents the tile within the FoV region with the greatest number of viewers. Specifically, we only show the popularity of the top 72 tiles here. The legend on the right side indicates the warmer the color (yellow) is, the more viewers, and the colder the color, the fewer viewers. The warm colors are concentrated on the few tiles with the highest attention on the left side, as shown in Fig. 1 (b). As expected, we observe that the viewer FoV is mainly focused on a small portion of all tiles. As shown in fig. 1 (c), the cosine similarity of head movement between two viewers with similar previous FoV (above 80% overlap) are distributed mainly in high-value intervals. For instance, the similarity value interval [0.8, 1] holds more than 87% population. Fig. 1 (d) shows the correlation coefficient distribution of the viewer pairs for all videos and reveal that the FoV coverage rate and the cosine similarity of the head movement are positively correlated.

The PLS system is a networked system in which multiple types of nodes are involved, including content providers, edge servers, and viewers with diverse devices. Providing low-latency and bandwidth-intensive PLS in such a complex networked system requires the design of an online, asynchronous, distributed, and cost-efficient solution in order to achieve high-quality viewer QoE. The time-shift phenomenon provides an opportunity to reuse existing FOV prediction models, and the similar behaviour makes tile reuse possible. These aspects have inspired us to design a new transmission framework that allows models and tiles be jointly reused to facilitate each other in PLS with the time-shift phenomenon. For this reason, we select clustered federated learning as the criterion to group viewers who are likely to reuse data and further provide a reinforced variational inference solution to assist other viewers in prediction model training and data transmission.

III. FedLive Framework and System Model

In this section, we first present the network model. We show the FoV prediction for tiled PLS, including the equirectangular projection and tile-based streaming. Finally, we introduce FedLife, the federated transmission framework for PLS.

The scalars are represented in lowercase *italic* symbols. Lowercase *italics* **bold** type and uppercase *italics* type indicate vectors and sets, respectively. Uppercase, double-strike **bold** fonts represent matrices. Table I lists all mathematical notations used in this paper.

A. Network Model

We consider the network topology of the PLS system as an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}|$ nodes and $|\mathcal{E}|$ links. Let $\mathcal{V}_p, \mathcal{V}_s, \mathcal{V}_c \in \mathcal{V}$ denote the set of content providers, edge servers and the viewers, respectively. We assume that the time is slotted in PLS and denoted by $\mathcal{T} = \{1, 2, ...\}$. Thus, the minimum interval of stream delay between different nodes is one slot. Since the network is subject to various cross-traffic from other applications, we define the available bandwidth resource of the link $e = (u, v) \in \mathcal{E},\ u, v \in \mathcal{V}$ at time t as a time-varying variable $c_e(t)$. Thus, we have $c_e(t) \in [0, c_e^{max}]$ where c_e^{max} is the maximum available bandwidth of link e. It is reasonable to assume that the content providers have sufficient computing capability to transcode the tile-level 360° video streaming and provide tile-based streaming with different standard resolutions in real time. The mainstream content providers such as YouTube, Netflix, etc., have their powerful own data center infrastructure to provide online transcoding for their rich video streaming content.

TABLE I
Mathematical Notations

Symbol	Description
$\mathcal{V}, \mathcal{E}$	The set of nodes and links of the network topology.
$\mathcal{V}_p, \mathcal{V}_s, \mathcal{V}_c$	The set of content providers, edge servers and viewers.
$\mathcal{T}$	A set of the time-slotted system
$c_e(t)$	The available bandwidth resource of the link e at time t.
$\mathcal{F}$	The set of different pieces of 360° video content.
$m, \mathbb{M}$	FoV prediction of viewer v.
$m^*, \mathbb{M}^*$	The ground truth FoV of viewer.
$\mathcal{B}$	The set of different resolutions for each tile.
$\mathbb{D}(v)$	The viewer's bitrate request for a specific frame.
$T^i_{u,v}(t)$	The t-slot shifted time of a node pair (u,v).
M, M_i	The set of local FOV records for all viewers (viewer i).
$R(\eta)$	Risk function
$R(s, a)$	Return function (total reward of all viewers)
$Q_\pi(s, a)$	Action-value function under policy π
$V_\pi(s, a)$	State-value function under policy π

B. FoV Prediction in Tiled PLS

We define the PLS library as set $\mathcal{F} = \{f_1, f_2, ..., f_{|\mathcal{F}|}\}$ which consist of $|\mathcal{F}|$ different pieces of 360° video content. Each video is divided into small video segments for transmission, and the video segment is further uniformly sampled into n frames. We define the set of frames of video f_i as $\mathcal{F}_i = \{f_i^1, ..., f_i^t, ...\}$. We assume that the PLS system processes the sphere video with equirectangular projection, and each video can be further divided into multiple rectangular tiles. Thus, the FoV prediction of viewer $v \in \mathcal{V}_c$ can be represented by a binary matrix $\mathbb{M}(v)$:

$$\mathbb{M}(v) = \begin{bmatrix} m_{1,1} & ... & m_{1,L} \\ ... & m_{k,l} & ... \\ m_{K,1} & ... & m_{K,L} \end{bmatrix}$$

where K and L denote the number of rows and columns, respectively, with $k \in [1, K]$, $l \in [1, L]$ and $m_{k,l} \in \{0, 1\}$.

We define $\mathbb{M}^*_v(t)$ as the ground truth requests of viewer v at time t. We denote the set of different resolutions for each tile as $\mathcal{B} = \{b_1, b_2, ..., b_{|\mathcal{B}|}\}$ where the bitrate of the highest resolution is b_1 and the lowest is b_m. The viewer's request for a specific frame at time t can be expressed as a matrix $\mathbb{D}(v)$:

$$\mathbb{D}(v) = \begin{bmatrix} d_{1,1} & ... & d_{1,L} \\ ... & d_{k,l} & ... \\ d_{K,1} & ... & d_{K,L} \end{bmatrix}$$

where $d_{k,l} \in \mathcal{B},\ \forall k, l$. We consider that the content provider and edge servers have all the tiles of the videos, while the viewers only have the tiles requested by themselves. Based on

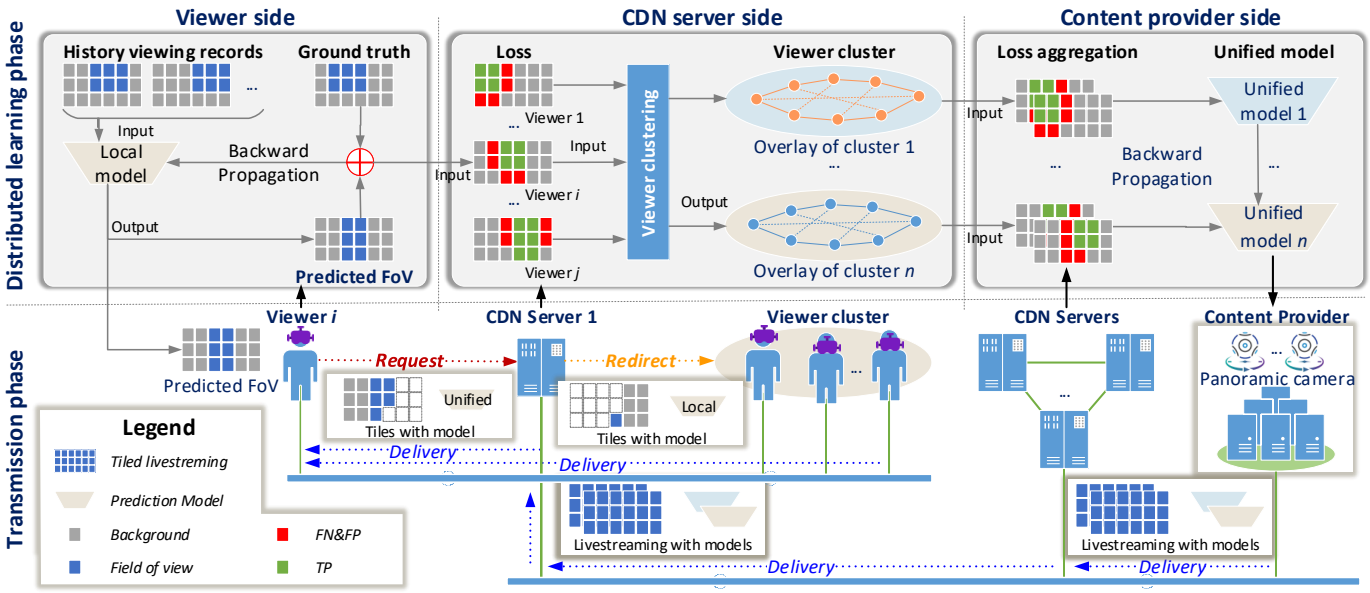


Fig. 2. Diagram of Federated Transmission Framework for PLS.

this information, the shifted time of any node pairs at a time t can also be calculated. We define the t-slot shifted time of a node pair $(u, v) \in \mathcal{V}$ for b_i-bitrate tile as $T^i_{u,v}(t)$.

C. FedLive Framework for PLS

Fig. 2 presents the diagram of FedLive, the proposed federated transmission framework for PLS. In FedLive multiple types of nodes are involved, including content provider, edge servers, and viewers with HMDs [38], [39]. FedLive contains two major parallel processes: federated learning for FoV prediction and content delivery, involving three blocks at viewers side, edge servers side and content provider side.

During federated learning, viewers train a local FoV prediction model with their local viewing records and saliency feature of video images. Based on the predicted results providing by the model, viewer determines the resolution of each prefetching tile for tile-based 360° video content. Note that the FoV prediction technique is the necessary foundation and prerequisite for the adaptive bitrate streaming (ABR) of 360-degree video. One feasible way is to predict the viewer's future region of interest by the FoV prediction model and then determine the resolution of tile streaming based on an ABR method [6], [22], [40]–[46].

During this training process, the loss information as output will be used for the local backward propagation and will also be employed by the nearby edge servers as input to the user clustering algorithm $(m^* + m)$. Specifically, the loss matrix is calculated by adding the predicted FoV binary matrix $\mathbb{M}$ to the ground truth binary matrix $\mathbb{M}^*$. Thus, the value of 2 in the loss matrix $(m^* + m)$ represents True-Positive (TP, tile that is actually viewed and is predicted to be within FoV) prediction results, 1 indicates False-Positive (FP, tile that is not actually viewed but is predicted to be within FoV) or False-Negative (FN, tile that is actually viewed but is predicted to be outside FoV), and 0 means True-Negative (TN, tile that is not actually viewed and is predicted to be outside FoV). Once the edge servers have collected the loss information from all viewers, they will invoke the gradient-based user clustering algorithm to divide the viewers into multiple viewer clusters.

At the same time, the edge servers continuously summarize the loss values for each cluster and send them to the content provider along with the clustering results as input for the unified model training. With the loss information provided by the edge servers, the content provider updates the unified model for each cluster with the weighted average loss value. This process extends the concept of federated learning [47] by adding clustering. Further, the unified models will be distributed to viewers as part of the live streaming, while the prefecthing priority of different tiles are determined as the predicted results of the unified models.

During the transmission process, the content provider first provides the edge servers the specific unified models and multi-resolution tiled 360° videos based on the clustering results submitted by the servers. Then, the edge servers distribute the unified models to viewers and support the tile-based live streaming according to viewers' requests. After receiving the unified model, viewers transform it into their local model and use it to predict the FOV. The prediction results are further used to determine the resolution of tiles in different regions. Specifically, *FedLive* can adapt to segment-based video transcoding and the system distributes the video content in segments. In addition, service demands may exceed the server's capacity with the increasing of access viewers.

In order to offload the traffic, in our framework, any edge server is allowed to redirect viewer requests to other edge servers or to the viewers who also access the server and are willing to offer their idle resources to support services. The proximal viewers belong to the same cluster of the requested viewer are preferred during redirection since they have the potential to provide not only the tiles but also the models. The viewers can access multiple alternative nodes, which may reduce the delay to get the demanded video content and

prediction model while saving bandwidth for the PLS system. Another important issue for livecast services is the online video transcoding. According to the previous assumptions in the network model, we consider that the content provider completes all the transcoding workload and streams the transcoded tile-based video content to the edge servers.

As the *FedLive* framework is focused on the federated learning of the FoV prediction and content delivery of 360° video streaming based on a prediction model, this manuscript does not include a discussion on transcoding issues and assumes that the content provider and edge servers provides video content at all resolutions required by viewers. In our previous work, we also proposed two joint optimization solution for data transmission and online transcoding, namely augmented queue model [48] and augmented graph model [49]. In the future, we will also consider the joint optimization of federated FoV prediction, content delivery, and transcoding task offloading in a FedLive network system.

IV. Gradient-Based Viewer Clustering and Reinforced Variational Inference for Federated Transmission Optimization

In this section, we first provide the necessary background about reinforced variational inference (RVI). Then, we give a brief introduction of clustered federated learning and gradient-based user clustering. Finally, we formulate the joint FoV prediction and tiled transmission problem in the RVI structure.

A. Background

1) Clustered federated learning [31]. This solution assumes that the data generating distributions of different users may be similar but disagreeing. Thus, finding only one single model that works for all users may not be an effective way in this situation. To facilitate an explanation, we first define the set of local FoV records for all viewers as $\{M_1, M_2, ..., M_{|\mathcal{V}_c|}\}$, which follows the data generating distributions $\{\psi_1, .., \psi_k\}$. We denote a parametrized function $\pi_\eta : \mathcal{X} \to \mathcal{Y}$ with η as the FoV prediction model and a loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$. We define a risk function $R(\eta)$ associated with ψ to evaluate the agreement of the parametrized function π_η and ψ:

$$R(\eta) = \int l\left(\pi_\eta\left(x\right), y\right) d\psi(x, y) \tag{1}$$

The parametrized function π_η is trained with viewer's viewing records whose distribution belongs to $\{\psi_1, .., \psi_k\}$. However, the data generating distributions is unknown for us, and so we cannot separate viewers by directly calculating the risk for each viewer. In this context, clustered federated learning provides a top-down way to recursively bi-partition the viewer population. Clustered federated learning defines an empirical risk function of viewer i as sampling results:

$$E_i(\eta) = \sum_{x \in D_i} l\left(\pi_\eta\left(x_i\right), y_i\right) \tag{2}$$

The separation criteria is defined in the next conditions:

$$\left\| \sum_{v_i \in \mathcal{V}_c} \frac{|D_i|}{|D|} \bigtriangledown_\eta E_i(\eta^*) \right\| < \xi_1 \tag{3a}$$

$$\max_{v_i \in \mathcal{V}_c} \left\| \bigtriangledown_\eta E_i\left(\eta^*\right) \right\| > \xi_2 \tag{3b}$$

where $|D| = \sum_{v_i \in \mathcal{V}_c} |D_i|$, and ξ_1, ξ_2 are the constants that satisfy $\xi_2 > \xi_1 > 0$. Inequation (3a) indicates that there is a stationary point for all distributions $\mathcal{D}$, and (3b) shows that there exits at least one viewer far from the stationary point. Namely, the overall gradient loss cannot represent the convergence of the local models for the individual viewers. We should conduct grouped training for viewers when the unified model converges but the local model still has a large gradient loss. The design of our gradient-based user clustering algorithm will follow the separation criteria.

2) Reinforced variational inference (RVI) [32]. The inference problem considers to find the posterior $p(y|x)$ under a environment represented by a probability model $p(y)p(x|y)$ with observations x and latent variables y. Since the exact posterior is often difficult to obtain, a common solution is to find an approximate distribution $q(y|x)$ that is close to the true posterior $p(y|x)$. In our problem, the approximate distribution $q(y|x)$ is represented by the parametrized function π_η for the FoV prediction, and the true posterior $p(y|x)$ follows one of the data generating distributions $\{\psi_1, ..., \psi_k\}$.

In general, the viewer i's viewing records can be represented as a set of sequences $M_i = \{\mathbb{M}_{i,1}, .., \mathbb{M}_{i,t}, ...\}$ where $\mathbb{M}_{i,t}$ denotes the ground truth FoV of viewer i at time t. $\mathbb{M}_{i,t}$ is what we want to predict, and represents the latent variables y in the inference problem. Thus, observation x denotes the input video frame f and historical viewing record $\mathbb{M}_{i,t-1}$. In variational inference (VI), the objective function $\mathcal{L}(q)$ is introduced to minimize the Kullback-Leibler (KL) divergence between the approximate distribution $q(y|x)$ controlled by π_η in our problem, and the true posterior $p(y|x)$.

$$\mathcal{L}(q) = \int q(y|x) \log \frac{p(x|y)p(y)}{q(y|x)} dy \tag{4}$$

where $p(x|y)$ is constant in our problem because the video frame is not affected by the viewer's viewpoint. We rewrite $q(y|x)$ as $q_\eta(y|x)$ for clarity. We consider FOV's inference process as a Markovian model and plug the parameters d and f into the objective function $\mathcal{L}(q_\eta)$. We can get the decomposed variational lower bound as follows:

$$\begin{aligned} \mathcal{L}(q_\eta) = \mathbb{E}\left[\log \frac{p(\mathbb{M}_1|f)}{q_\eta(\mathbb{M}_1|f)} + \log \frac{p(\mathbb{M}_2|\mathbb{M}_1, f)}{q_\eta(\mathbb{M}_2|\mathbb{M}_1, f)} + ... + \right. \\ \left. \log \frac{p(\mathbb{M}_T|\mathbb{M}_{T-1}, f)}{q_\eta(\mathbb{M}_T|\mathbb{M}_{T-1}, f)}\right] = \mathbb{E}\left[\sum_{t=1}^{T} -r_t(\mathbb{M}_t, \mathbb{M}_{t-1}, f)\right] \end{aligned} \tag{5}$$

where $r_t(\mathbb{M}_t, \mathbb{M}_{t-1}, f) = -\log\left(\frac{p(\mathbb{M}_t|\mathbb{M}_{t-1})}{q_\eta(\mathbb{M}_t|\mathbb{M}_{t-1}, f)}\right)$. Thus, we can express VI as a 4-tuple Markov decision process where

- State $S : s_t \triangleq (\mathbb{M}_{t-1}, f_t)$. For our problem, the state s_t at time t includes the viewer's viewing record $\mathbb{M}_{t-1}$ at previous time-slot and the video frame f_t at slot t.

- Action $A: a_t \triangleq \mathbb{M}_t \sim q_\eta(\mathbb{M}_t|\mathbb{M}_{t-1}, f_t)$. The action a_t at time t represents the viewpoint records $\mathbb{M}_t$ at time t based on the posterior q_η, which can be regarded as the viewpoint movement according to the frame changes.
- Transition $P(\mathbb{M}'|\mathbb{M}, f) \triangleq (\mathbb{M}', f) \sim ((\mathbb{M}, f), \mathbb{M}') \rightarrow [0, 1]$. The transition P is a probability function from the state $(\mathbb{M}, f)$ to $(\mathbb{M}', f)$ when the action $a = d'$ is taken.
- Return $R \triangleq \sum_{v\in\mathcal{V}_c} -\mathcal{L}(q_\eta)$. The return is the total reward $\mathcal{R}$ of all viewers which is denoted as

$$R(\boldsymbol{s}, \boldsymbol{a}) = \sum_{v\in\mathcal{V}_c}\sum_{t\in\mathcal{T}} \mathbb{E}[r_t^v|\boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}] \tag{6}$$

where r_t^v is the instant reward of viewer v at time t.

The policy π is a mapping $S \times A \rightarrow [0, 1]$ and is denoted by the parametrized function π_η in our problem.

Thus, different users' reward functions are independent, allowing us to design a distributed Federated learning solution in the asynchronous network system. Based on a standard regularity assumption, we assume that the Markov chain of VI Markov decision process is irreducible and aperiodic [50]. The state-value function $V_\pi(\boldsymbol{s})$ and the action-value function $Q_\pi(\boldsymbol{s}, \boldsymbol{a})$ can be expressed as follows:

$$Q_\pi(\boldsymbol{s}, \boldsymbol{a}) = \sum_{v\in\mathcal{V}_c} \mathbb{E}\left[\bar{r}_{t+1} - R_v(\pi, t)|\, s_0,\, a_0,\, \pi\right] \tag{7a}$$

$$V_\pi(\boldsymbol{s}) = \sum_{a\in A} \pi(\boldsymbol{s}, \boldsymbol{a}) Q_\pi(\boldsymbol{s}, \boldsymbol{a}) \tag{7b}$$

where $\bar{r}_{t+1} = \frac{1}{|\mathcal{V}_c|}\sum_{v\in\mathcal{V}_c} r_{t+1}^v$. With Bellman equations, we can design multi-types RL algorithms, such as actor-critic (AC) [51] or Q-learning [52], to solve the inference problems. Since the PLS system is asynchronous, the AC algorithm is an effective alternative solution to achieve collaborative learning between viewers.

B. Gradient-Based Viewer Clustering (GVC)

When employing clustered federated learning in the *FedLive* framework, we need to consider the workflow of the user clustering. The PLS system start when the first viewer requests a specific video from content provider. The request will be captured by the nearby edge server. Next, the edge server provides the viewer with the requested live streaming and a unified model. The viewer will use the unified model as the local model to predict FoV and upload the gradient loss to the PLS system. In the meanwhile, the viewer can choose whether to train the local model or just wait for the next unified model produced by the content provider. As the number of access viewers increases, the edge servers will collect the empirical risk $E_i(\eta)$ updated by the accessed viewers and calculate the separation criteria with the unified model $\pi_{\eta^*}(x)$ according to (2). If the separation criteria is met, the edge server will first group the viewers and transmit the loss gradient to the content provider in groups. The similarity between any two viewers with the same unified model is represented as the cosine similarity $\beta_{i,j}$ of their loss gradient $\bigtriangledown_\eta E_i(\eta^*)$ as follows:

$$\beta_{i,j}\left(\bigtriangledown_\eta E_i(\eta^*), \bigtriangledown_\eta E_j(\eta^*)\right) \triangleq \frac{\langle \bigtriangledown_\eta E_i(\eta^*)\, \bigtriangledown_\eta E_j(\eta^*)\rangle}{\parallel \bigtriangledown_\eta E_i(\eta^*) \parallel \parallel \bigtriangledown_\eta E_j(\eta^*) \parallel}$$

When the content provider receives grouped gradient loss from multiple edge servers, it trains a unified model for each group of each server, respectively. Then, the content provider decides whether to merge some groups by comparing the cosine similarity of the parameters η of different unified models. Here, we set a threshold ξ_0 at which the two different groups will be merged when the cosine similarity is greater than ξ_0. At this point, the user clustering is complete, and the content provider delivers the tile-based video content and unified models to the edge server based on the grouping.

C. Problem Formulation based on the RVI Structure

Our goal is to provide high-accuracy prediction model for the viewers while optimizing data transmission. As previously mentioned, the collaborative learning of prediction model can be converted into a variational inference problem that aims at minimizing the KL divergence. Therefore, here we focus on data transmission and how to formulate the joint optimizations.

We define the received data rate of viewer i for tile j at time t as $\alpha_j^i(t)$. Thus, the utility of transmission for PLS is $\sum_{i\in\mathcal{V}_c}\sum_{j\in\mathcal{M}} \mathcal{U}(\alpha_j^i(t))$ where $\mathcal{U}(\cdot)$ is the utility function and $\mathcal{M}$ is the set of tiles. We can use the form employed in [53], as the utility function. A standard way to formulate the optimization is as follows:

$$\forall t: U(t) = \max\left\{\sum_{i\in\mathcal{V}_c}\sum_{j\in\mathcal{M}} \mathcal{U}\left(\alpha_j^i(t)\right)\right\} \tag{8a}$$

$$\text{s.t.} \sum_{i\in\mathcal{V}_c(e)}\sum_{j\in\mathcal{M}} \alpha_j^i(t) \le c_e(t),\ \forall e \in \mathcal{E} \tag{8b}$$

$$\alpha_i(t) \in \left[0, b_{|\mathcal{B}|}\right],\ \forall i \in \mathcal{V}_c \tag{8c}$$

where $\mathcal{V}_c(e)$ is the set of viewers that their video flow pass through the link e. Eq. (8a) is the objective function that maximizes the total utility. The constraint in eq. (8b) indicates that the data rate cannot exceed the available link's bandwidth and (8c) is the boundary constraint. The optimization problem is separable about both the viewers and the tiles, and the same is true for the inference problem. Because the probability of whether a region is in the FoV can be divided into a product of the probabilities of whether the tile is in the FoV. This can be expressed as follows:

$$p\left(\mathbb{M}'|\mathbb{M}\right) = \prod_{k\in[1,K], l\in[1,L]} p\left(m'_{k,l}|\mathbb{M}\right) \tag{9}$$

Based on eq. (5), (8a)-(8c), and (9), we formulate the joint optimizations of data transmission and VI as in the following unconstrained expression:

$$\begin{aligned}\forall t: U'(t) = \max\Bigg\{&\sum_{i\in\mathcal{V}_c}\sum_{j\in\mathcal{M}} \lambda\log\left(\mathcal{U}\left(\alpha_j^i(t)\right.\right.\\ &\left.\left.-\phi\left(\boldsymbol{\alpha}(t) - c_e(t)\right)\right) - \log\frac{p(m_j(t)|\mathbb{M}_{t-1}(i))}{q_\eta(m_j(t)|\mathbb{M}_{t-1}(i), f)}\Bigg\}\end{aligned} \tag{10}$$

where $\boldsymbol{\alpha}(t) = \sum_{i\in\mathcal{V}_c(e)}\sum_{j\in\mathcal{M}} \alpha_j^i(t)$ and ϕ, λ are the weight factors. The objective function consists of two parts: the

previous term can be regarded as a LOG Lagrange function $G_j^i(\boldsymbol{\alpha}(t)) = \lambda \log\left(\mathcal{U}\left(\alpha_j^i(t)\right) - \phi\left(\boldsymbol{\alpha}(t) - c_e(t)\right)\right)$ for the transmission optimization. The second is the instant reward at time t of RVI. For each tile, the probability of being viewed by the viewer i is either 0 or 1. Thus, the second term can only be ∞ or $\log \frac{1}{q_\eta(m_j(t)|\mathbb{M}_{t-1}(i),f)}$. In other words, when the tile is not in the viewer's FoV, the reward is dominated by the second term, and no matter how we increase α, we can not get a high reward. Increasing the transmission rate of tile outside the FoV may occupy the link bandwidth and decrease the overall reward. Oppositely, when tiles are within the FoV, increasing the transmission rate within the bandwidth capacity can improve the reward. Thus, we can reformulate the joint optimization problem in the RVI structure as the following 4-tuple Markov decision process.

- State $S' : s_t \triangleq (\mathbb{D}_{t-1}, f_t, \boldsymbol{c}_t)$. $\mathbb{U}'(t-1) = G(\mathbb{D}_{t-1})\mathbb{M}_{t-1}$ is the utility matrix of the tiles requested at time $t-1$. $\boldsymbol{c}_t$ indicates the set of available bandwidth for all links $\mathcal{E}$ at time t.
- Action $A' : a_t \triangleq \mathbb{D}_t \sim q_\eta(\mathbb{D}_t|\mathbb{D}_{t-1}, f_t, \boldsymbol{c}_t)$.
- Transition P' : $P'(\mathbb{D}'|\mathbb{D}, f, \boldsymbol{c}_t) \triangleq (\mathbb{D}', f, \boldsymbol{c}_t) \sim ((\mathbb{D}, f, \boldsymbol{c}_t), \mathbb{D}') \to [0, 1]$.
- Return $R'(\boldsymbol{s}, \boldsymbol{a}) \triangleq \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{V}_c} \sum_{j\in\mathcal{M}} R_j^i(q_\eta)$ where:

$$R_j'^i(q_\eta) = \begin{cases} \log\left(G_j^i\left(\boldsymbol{\alpha}\right) q_\eta\left(d_j|\mathbb{D}\left(i\right), f\right)\right) & m_j = 1 \quad (11a) \\ 0 & m_j = 0 \quad (11b) \end{cases}$$

For simplicity, we let $R_j'^i(q_\eta) = 0$ when $m_j = 0$ and rewrite $\bar{r}_{t+1} = \frac{1}{|\mathcal{V}_c|} \sum_{v\in\mathcal{V}_c} r_{t+1}'^v$ where $r_{t+1}'^v = \sum_{j\in\mathcal{M}} R_j^i(q_\eta)$. Thus, we have $R_v'(\pi, t) = \mathbb{E}_d\left[\sum_{\tau=1}^t r_\tau'^v(\mathbb{D}_\tau, \mathbb{D}_{\tau-1}, f_\tau)\right]$, and the following forms of the action-value function $Q_\pi(\boldsymbol{s}, \boldsymbol{a})$ and the state-value function $V_\pi(\boldsymbol{s})$.

$$Q_\pi\left(\boldsymbol{s}, \boldsymbol{a}\right) = \sum_{v\in\mathcal{V}_c} \mathbb{E}\left[\bar{r}_{t+1}' - R_v'\left(\pi, t\right) | \, s_0, \, a_0, \, \pi\right] \quad (12a)$$

$$V_\pi(\boldsymbol{s}) = \sum_{a\in A} \pi(\boldsymbol{s}, \boldsymbol{a}) Q_\pi(\boldsymbol{s}, \boldsymbol{a}) \quad (12b)$$

V. Algorithm Design

In this section, we introduce the gradient-based viewer clustering algorithm. Then, we give the prediction-based asynchronous delivery algorithm based on RVI to solve the problem (10) and describe the implementation of the algorithms.

A. GVC Algorithm

We first provide the pseudo-code (**Algorithm 1**) of gradient-based viewer clustering algorithm (GVC) which is deployed at the edge servers $\mathcal{V}_s$ and runs at regular intervals T_i. The servers take the set of accessed viewers' set $\mathcal{V}_\mathrm{I}$, the collected gradient loss $\{\bigtriangledown_\eta E_i(\eta^*)\}$, and the calculated cosine similarity $\boldsymbol{\beta}_{\mathcal{V}_\mathrm{I}}$ as inputs of the algorithm. First, the algorithm sorts cosine similarity vectors in descending order and stores the index vector of viewer pairs in vector $V[:, 2]$. The algorithm then takes the elements of $V[:, 2]$ in sequence. Before separating the set $\mathcal{V}_\mathrm{I}$, the algorithm will first judge whether all the viewer sets in $\mathcal{V}_\mathrm{I}$ meet the separation criteria. If the criteria is met, the algorithm continues to execute, otherwise, exits and updates data $\mathcal{V}_\mathrm{I}$ and $\{\bigtriangledown_\eta E_i(\eta^*)\}$, etc. to content provider directly. Otherwise, the algorithm splits the $\mathcal{V}_\mathrm{I}$, first splitting the two most similar elements into a new viewer set $\{\mathcal{V}_\mathrm{T}\}$, and iterating the process in turn. The algorithm stops and uploads data until none of the viewer sets in $\mathcal{V}_\mathrm{I}$ satisfies the separation criteria. The algorithm ends up with bipartition. If the bipartition still satisfies the criteria, we take the bipartition as the new input, and execute the algorithm.

Algorithm 1: Gradient-based Viewer Clustering

Input: Set of viewer' set $\mathcal{V}_\mathrm{I} = \{\{i\}|i \in \mathcal{V}_{v_s}\}$, gradient loss vector $\{\bigtriangledown_\eta E_i(\eta^*)\}, i \in \mathcal{V}_\mathrm{I}$, the similarity vector $\boldsymbol{\beta}_{\mathcal{V}_\mathrm{I}} = \{\beta_{i,j}\}^{\mathrm{C}^2_{\mathcal{V}_\mathrm{I}}}, i, j \in \mathcal{V}_\mathrm{I},\ i \neq j$.

```
V[:,2] ← argsort(β_{V_I}) ∈ (i,j)^{C²_{V_I}}
for n = 1,2,...,C²_{V_i} do
    foreach v ∈ V_I do
        if v satisfies the criteria (3a),(3b) then
            Break
        else if v = V_I[end] then
            Return V_I
        end
    end
    V_T ← {}
    foreach v ∈ V_I do
        if V[n,0] ∈ v or V[n,1] ∈ v then
            V_I ← V_I \ v; V_T ← V_T ∪ v
        end
    end
    V_I ← V_I ∪ {V_T}
end
```

B. Prediction-Based Asynchronous Delivery Algorithm

Algorithm 2 shows the prediction-based asynchronous deliver algorithm (PAD). First, we initialize policy π_{η_0} (the unified model) with parameter η_0 at content provider and set the non-negative stepsize γ_η. The content provider distributes the unified model π_{η_0} to the edge servers and keeps the model synchronized during the whole period. When the viewers start the request, the policy π_{η_0} is periodically delivered to the viewers along with the 360° video content.

The algorithm can be divided into three parts: at the viewer, server, and provider, respectively. At the viewer side, the viewer u downloads the unified model π_{η_t} at interval T_u to update the local model $\pi_{\eta_t}^u$. The model integrates the experience of other preceding viewers in the asynchronous delivery system, which improves the model's prediction performance under the time-shift scenario. the algorithm uses the local policy $\pi_{\eta_t^u}$ to predict the viewer's FoV $a_t^u = \mathbb{D}_t'$. Here, the viewer is free to update the local policy $\pi_{\eta_t^u}$ or just upload the information, such as $E_u(\eta_t^u)$ and $\bigtriangledown_\eta E_u(s_t, a_t)$, to the edge server and get the new policy π_{η_t} provided by content provider. According to eq. (10) and $\mathbb{U}_t$=$\mathbb{G}(\mathbb{D}_t)\mathbb{M}_t$, the loss is the difference between the actual utility a_t^u of viewer u and the optimal utility $\mathbb{D}_t^*$ which can be calculated based on the

ground truth viewing. The gradient loss $\bigtriangledown_\eta E_u(s_t, a_t)$ can also be calculated based on eq. (10).

The edge servers are responsible for viewer clustering. The servers use $\{\bigtriangledown_\eta E_u(s_t, a_t)\}_{\mathcal{V}_I}$ to calculate $\beta_{\mathcal{V}_I}$, then employ the GVC to get the grouping $\mathcal{V}_I$, and finally upload data $\mathcal{V}_\text{I}$, $\{E_u(\eta_t^u)\}$, and $\{\bigtriangledown_\eta E_u(s_t, a_t)\}$ by cluster. The content provider performs standard policy gradient updates. After receiving the information $\{E_u(\eta_t^u)\}$, and $\{\bigtriangledown_\eta E_u(s_t, a_t)\}$ of a cluster $\mathcal{V}_\text{I}$, the algorithm execute the updates by first calculating the expectation of historical returns $R'(\pi_\eta, t)$. We can further calculate the joint action value Q_t of $\mathcal{V}_\text{I}$ by using $R'(\pi_\eta, t)$ minus $\sum_{u\in\mathcal{V}_i} E_u/|\mathcal{V}_i|$. Since $\sum_{u\in\mathcal{V}_i} E_u/|\mathcal{V}_i|$ is the average reward at time t, we consider it as the approximation of the expected return if the policy π_{η_t} is unchanged. Our goal is to get policy close to utility (10), thus step 17 of the algorithm can be viewed as the score $\bigtriangledown_\eta \log \pi_\eta(s, a)$ [51]. Similarly, the advantage value is the difference between the action and state values (eq. (12b)). With the advantage value, the policy parameters η_{t+1} are updated and the algorithm distributes the new unified policy $\pi_{\eta_{t+1}}$ to the viewers.

C. Implementation of the Algorithms

Algorithm 1 is a central synchronization algorithm which is deployed at the edge servers. Its time complexity is $\mathcal{O}(|\mathcal{V}|^3)$ and space complexity is $\mathcal{O}(|\mathcal{V}|^2)$ where $|\mathcal{V}|$ is the number of the accessed viewers. **Algorithm 2** is a distributed asynchronous online algorithm that aims at enhancing the FoV prediction and data transmission through collaborative learning between the viewers. The asynchronism is reflected in that the viewers can asynchronously update data at the edge server, and likewise, the edge servers - at content provider. The model is iteratively updated, and every node participates, so it is online and distributed. The space complexity and time complexity of **Algorithm 2** are both $\mathcal{O}(|\mathcal{V}|)$ if we leave out the server-side's **Algorithm 1**. To prove algorithms' feasibility, we also evaluate the signaling bandwidth overhead in Section VI.

Algorithm 2: Prediction-based Asynchronous Delivery

Input: Random η_0 and initial policy π_{η_0} at content provider; set γ_η as the nonnegative stepsizes.

while $t \in \mathcal{T}$ **do**
  foreach *Viewer* $u \in \mathcal{V}_c$ **do**
    if $(t \bmod T_u) == 0$ **then**
      obtains the policy model π_{η_t} from the servers as the local model $\pi_{\eta_t^u}$.
    end
    selects the action $a_t^u \leftarrow \pi_{\eta_t^u}(s_t, \cdot)$;
    obtains the next state $s_{t+1} \leftarrow (\mathbb{D}_t, f_\tau, \boldsymbol{c}_{t+1})$;
    gets the loss $E_u(\eta_t^u) \leftarrow a_t^u - \mathbb{D}_t^*$ and train $\pi_{\eta_t^u}$;
    calculates $\bigtriangledown_\eta E_u(s_t, a_t)$ and upload;
  end
  foreach *edge server* $v \in \mathcal{V}_s$ **do**
    calculates $\beta_{\mathcal{V}_I}$ with $\{\bigtriangledown_\eta E_u(s_t, a_t)\}_{\mathcal{V}_I}$;
    $\mathcal{V}_\text{I} \leftarrow \text{GVC}\left(\mathcal{V}_\text{I}, \{\bigtriangledown_\eta E_u(s_t, a_t)\}, \beta_{\mathcal{V}_\text{I}}\right)$;
    uploads $\mathcal{V}_\text{I}$, $\{E_u(\eta_t^u)\}$, and $\{\bigtriangledown_\eta E_u(s_t, a_t)\}$;
  end
  /* Content provider */
  foreach $\mathcal{V}_i \in \{\mathcal{V}_\text{I}\}_{\mathcal{V}_s}$ **do**
    $R'(\pi_\eta, t) \leftarrow \frac{t-1}{t} R'(\pi_\eta, t-1) + \frac{1}{t} \frac{\sum_{u\in\mathcal{V}_i} E_u}{|\mathcal{V}_i|}$;
    $Q_t \leftarrow \mathbb{E}\left[\frac{\sum_{u\in\mathcal{V}_i} E_u}{|\mathcal{V}_i|} - R'(\pi_\eta, t)\right]$;
    $\phi_t \leftarrow \frac{1}{|\mathcal{V}_i|} \sum_{u\in\mathcal{V}_i} \bigtriangledown_\eta \log E_u$;
    $A_t \leftarrow Q_t - \sum_{a\in A} \pi_\eta(s, a) \cdot [E_u - R'(\pi_\eta, t)]$;
    $\eta_{t+1} \leftarrow \eta_t + \gamma_\eta \cdot A_t \cdot \phi_t$ and distribute $\pi_{\eta_{t+1}}$;
  end
end

VI. Performance Evaluation

In this section, we introduce the experimental settings and dataset. We analyse the results of our proposed prediction-based asynchronous delivery algorithm and evaluate its performance by comparing it with that of three state-of-the-art solutions: *PanoSalNet* [8], *LiveDeep* [14], and *LiveObj* [15].

A. Experiment Setup

We implemented the system simulation architecture for PLS illustrated in the diagram shown in Fig. 3, by adapting **PyTorch**[4] with Python 3.8. The system simulates a scenario with one content provider, two edge servers, and 48 virtual viewers. The content provider and two edge servers form together a simple CDN. We deployed the system-level simulation on a full-fledged laptop (AMD R9 5900HX, 32G, RTX 3080). The source code used is available in a GitHub repository[5]. To simulate the tiled mechanism of panoramic video, we developed a visual playground for FoV prediction based on the OpenAI-Gym[6] library, which converts 360° videos into tile-based (5 tiles × 5 tiles) streaming with equirectangular projection. According to a HUAWEI white paper [2], the 360° video full-view resolution for good quality of experienceis at least 8K (7680*4320). Since each full-view frame is partitioned into 25 tile regions, we consider splitting the panoramic video into 25 tile-based video streams, where each video stream corresponds to a region. The HEVC tiling toolkit[7] is employed for tiled video data preparation and the real-time messaging protocol (RTMP) over the User Datagram Protocol (UDP)-multicast[8] is used to transfer 25 live tiled video streams. The tile streaming contains six standard resolutions, and we set up the bandwidth requirement for different resolutions, as shown in Table II. Table II results are measured with an Amazon Web Services instance and Twitch's official tool according to [54]. We initialized the parameters $\eta_0 \in (0, 1)^N$ with uniform distribution, $\xi_0 = \xi_2 = 0.5$, $\xi_1 = 0.1$, $\phi = 10$, $\lambda = 1$, and $\gamma_\eta = 3 \times 10^{-4}$. We set the interval of local model updates for all viewers T_u, $\forall u \in \mathcal{V}_c$ to 30s. We employed the Boltzmann exploration and set the policy function π_η as follows:

[4]https://github.com/pytorch/pytorch

[5]hhttps://github.com/uglyghost/FedLive

[6]http://gym.openai.com/

[7]https://github.com/ultravideo/kvazaar

[8]https://openwrt.org/de/docs/guide-user/network/wan/udp_multicast

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2023.3237325

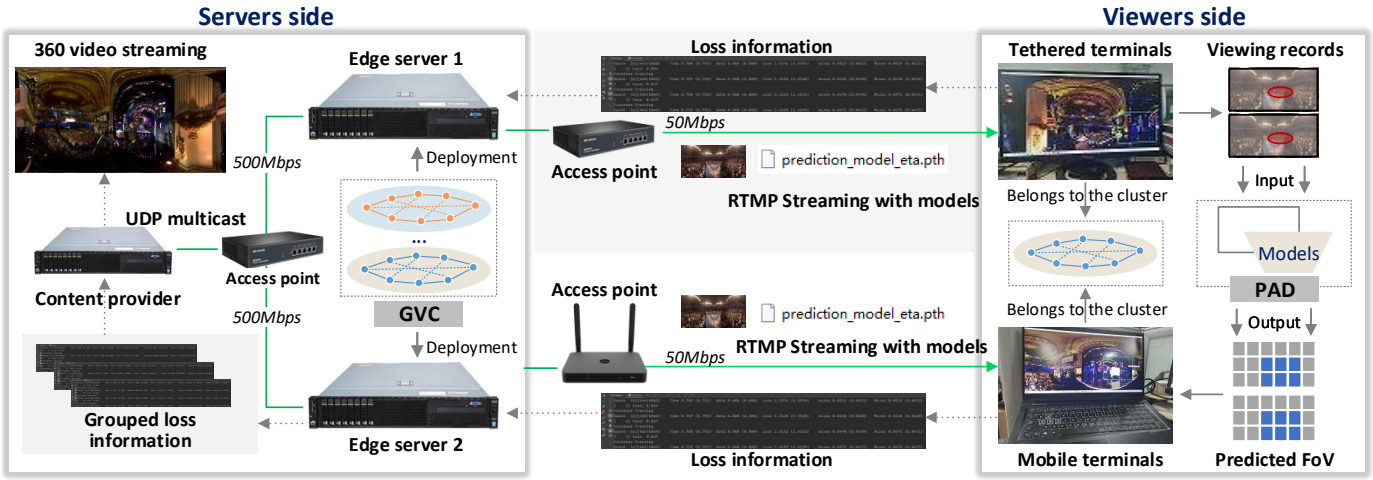


Fig. 3. The topology diagram of the system-level simulation.

TABLE II
BANDWIDTH REQUIREMENTS PER TILE FOR DIFFERENT RESOLUTIONS

resolution	1080p 60fps	1080p	720p 60fps	720p	480p	360p
bandwidth (Mbps)	5.86	4.45	2.75	1.93	1.10	0.52

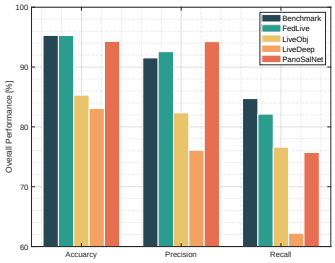


Fig. 4. Overall performance of four solutions (accuracy, precision, recall).

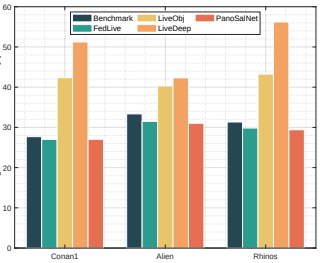


Fig. 5. The number of selected tiles with high resolution for four solutions.

$$\pi_\eta(a|s) = \frac{\exp\left(\mathcal{Q}_\eta(s,a)\right)}{\sum_{a' \in A} \exp\left(\mathcal{Q}_\eta(s,a')\right)} \tag{13}$$

where $\mathcal{Q}_\eta$ is a function in the form of eq. (10) with parameterized function q_η. The state s provides $\boldsymbol{c}_t$ and $\mathbb{D}_{t-1}$ as inputs, $p(m|\mathbb{M})$ is based on the sampling results, and $q_\eta \triangleq \omega^\top \eta$ where $\omega^\top$ is the feature vector for a specific action.

Various reinforcement learning techniques, such as actor-critic (AC) [51] or Q-learning [52], can be applied to our framework. To meet the real-time requirements of live-cast services, we design the reinforced variational inference model based on the classical actor-advantage-critic method. The model consists of four neural network architectures: actor-network, critic-network, value-network, and target value-network, and each of the networks uses the multilayer perceptron (MLP) structure with two hidden layers. We set the sampling interval of video frames to 8 and each video segments (2 seconds with 30fps) can be extracted into 8 frames. Each video frame is divided into 5×5 regions, represented by the matrix $\mathbb{M}$. For actor-network and critic-network, we set the size of the input layer to 400, where the input vector contains the information of both saliency features and the user's viewing records for 8 consecutive sampled frames. The output layer contains 200 neurons which represent the predicted user's viewpoint $\mathbb{M}$ for the next 2 seconds of the future video segment.

Furthermore, we discuss some essential details of the dataset and show how it contributes to our experiments. **Spherical Head Tacking Dataset** [34]: This dataset provides the head trajectories of 48 users (24 males and 24 females) as they watched 18 different panoramic videos from different categories including speeches, sports, and competitions. The dataset includes users' head movement in each session, impressive targets of each user in the panoramic video, and users' head orientations in terms of the unit quaternion (X, Y, Z, W) and HMD position (x, y, z) in the panoramic video. We transformed the dataset's records to a corresponding saliency map and viewport following the rules provided in [37].

B. Simulation Workflow

In this section, we describe our simulation workflow based on the topology diagram shown in Fig. 3. We set three content servers in the system-level simulation, including one content provider and two edge servers. We consider that the content provider is responsible for online video transcoding, and distributing the processed 360-degree video to the edge servers via 500Mbps wire link. The edge server directly responds to the viewers' requests and communicate with each viewer through 50Mbps wire link. As indicated, we used UDP-multicast-based RTMP to implement tile-level live 360° video streaming among the content provider, edge servers, and viewers in a real-world application.

We also deployed the unified model on the servers and keep them synchronized during the whole period of the experiment. The simulation is driven by having viewers request the video content in sequence. In the initial phase, the edge servers service the viewers with low definition tile-level live 360° video streaming and the unified model. The first requested

viewer experiences a slow start process and upgrades the downloaded unified model as the local model. Then, the viewer prefetches the high-resolution tiles based on the prediction results of the model. The viewer trains the local model based on the gap between the predicted FoV and the ground truth FoV and uploads the loss information to the servers. The content provider updates the unified model with the loss information and streams the updated unified model to the edge servers. Then, the edge servers respond to the subsequent viewers with the updated model. We achieved the time-shift phenomenon by configuring the viewers to start playing from the beginning of the video while they have various startup times. Thus, the predicted model was trained with the viewing records of the foregoing viewers, and was directly applied for the FoV prediction of the subsequent viewers. When the edge servers collect loss information from multiple viewers, they invoke the gradient-based user clustering algorithm to divide viewers into multiple viewer clusters. The system maintains one unified model for each cluster, and the loss of each viewer in the cluster is used for model training.

C. Evaluation of Results

We consider the centralized synchronous RVI-structure algorithm as the **Benchmark**. In this case, the PLS system contains only content provider and viewers without the shifted time, and the algorithm is deployed at the content provider side with global information. First, we evaluated the prediction performance of our solution in terms of accuracy, precision, and recall and we compared it with that of three state-of-the-art solutions: *PanoSalNet* [8], *LiveDeep* [14], and *LiveObj* [15] in an asynchronous PLS system. Further, we compared the proposed solution's service performance and resource efficiency with the other three solutions.

- **PanoSalNet** is an offline solution that predicts the FoV with 360 feature extraction and multi-modal learning. The authors first design an attention-based viewport prediction framework based on the extracted saliency features. Then, a mixture density network-based viewport predictor and a matching spherical loss are proposed to accelerate the training.
- **LiveDeep** is a online training solution which deploys the prediction model at the server side. The viewers need to upload the loss to the server. LiveDeep uses a CNN to capture the image features and an LSTM to analyze viewers' viewing trajectory. It predicts viewer's viewport based on these two results.
- **LiveObj** is similar to *LiveDeep* in terms of the deployment. It formulates the FoV prediction problem in a RL structure, where the state is represented by the coverage of the object and viewport in the tiled image, and the action is select a tiled FoV as prediction result.

We evaluated the predicted performance of our solution in terms of FOV prediction accuracy, precision, recall and bandwidth consumption (the number of prefetch tiles). Specifically, accuracy represents the proportion of all 360-degree video tiles that are correctly predicted which can be represented as equation $\mathrm{accuracy} = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FN+FP}}$. Precision represents the ratio of tiles viewed and correctly predicted to all tiles actually viewed by viewers ($\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$). Recall represents the ratio of tiles viewed and correctly predicted to all predicted tiles ($\mathrm{recall} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}$). Fig. 4 demonstrates the overall predicted performance of four solutions with a central benchmark for 9 different videos and 48 different viewers. For accuracy, *FedLive* and Benchmark provide optimal average performance when compared to the other three solutions and improve the performance by almost 1 percent when compared to *PanoSalNet*. However, *PanoSalNet* outperforms all other solutions, including *FedLive*, in terms of precision with about 1 percent. Furthermore, our approach has advantages over both *LiveDeep* and *LiveObj* in accuracy, precision, and recall. One significant reason why *FedLive* is superior to *LiveDeep* in asynchronous PLS system is that our solution can improve the prediction performance by utilizing multiple viewers to train the prediction model cooperatively. Recall that our solution provides the best performance, higher than that of *PanoSalNet* by about 5%. Although *PanoSalNet* achieves comparable predicted performance compared to our proposed solution, *PanoSalNet* requires an offline training process and is difficult to be applied to panoramic livecast services. Fig. 5 illustrates the average number of selected tiles for the four solutions under 3 different types of videos. Since both *Fedlive* and *PanoSalNet* achieve excellent predicted results, these two solutions save bandwidth resources by prefetching fewer high-resolution tiles to viewers. *Livedeep* has the worst performance because it takes the largest number of high-definition tiles, many out of the viewing area as selected tiles.

To support the above results, we provided more detailed experimental results in the following analysis. Fig. 6 gives more details of the four solutions on the accuracy performance. The results show that the asynchronous *FedLive* can approximate the accuracy performance of the synchronous Benchmark with an acceptable degradation. The accuracy of *FedLive* remains above 90% for all the videos being more stable when compared to other solutions. In addition, *PanoSalNet* also achieves no less than 90% accuracy for all the videos while *LiveDeep* and *LiveObj* have 78%-90% and 81%-90% accuracy, respectively. Fig. 7 presents the average precision performance of the four solutions. *FedLive* and *PanoSalNet* achieve around 90% precision for all the videos, and our solution has the best performance in War, Cooking, and Football. In addition, the precisions of *LiveDeep* and *LiveObj* are between 63%-88% and 65%-91%, respectively. Fig. 8 shows the average recall of the four solutions. Since *LiveDeep* and *LiveObj* over-prefetch involving a higher number of high-definition tiles outside the field of view, their performance of recall decreased significantly. Our approach significantly outperforms all other solutions on recall performance for most of the videos except Football for which *PanoSalNet* has the highest recall. In conclusion, our solution not only outperforms other solutions in terms of accuracy precision, and recall, but also has a more stable performance for different types of videos.

Fig. 9 gives details about the predicted results with our

This article has been accepted for publication in IEEE Transactions on Multimedia. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMM.2023.3237325

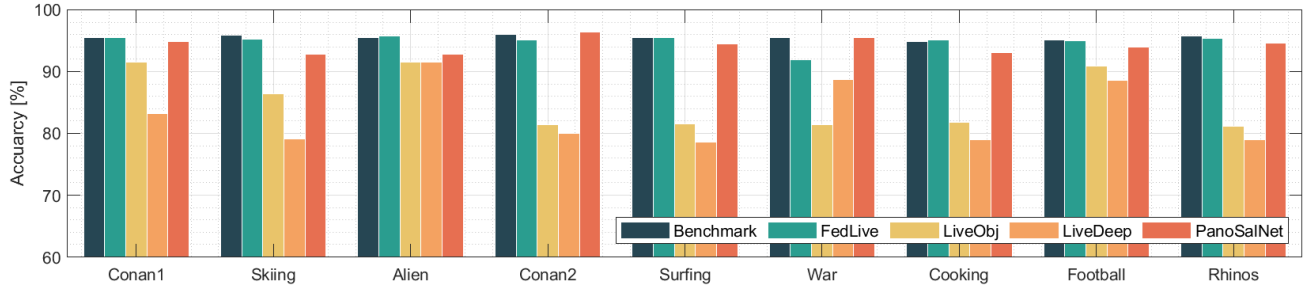


Fig. 6. The average accuracy of different videos for four solutions.

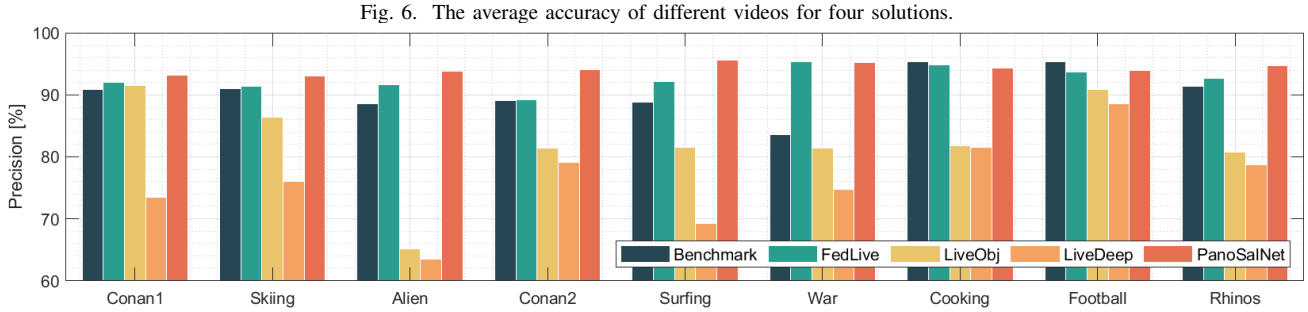


Fig. 7. The average precision of different videos for four solutions.

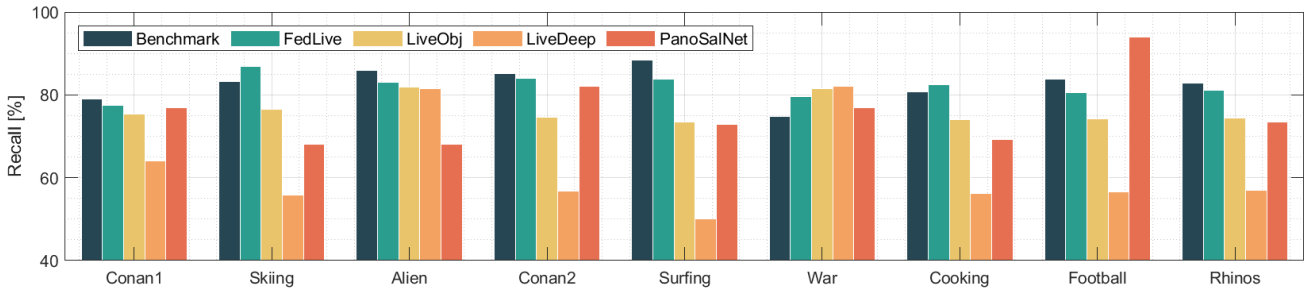


Fig. 8. The average recall of different videos for four solutions.

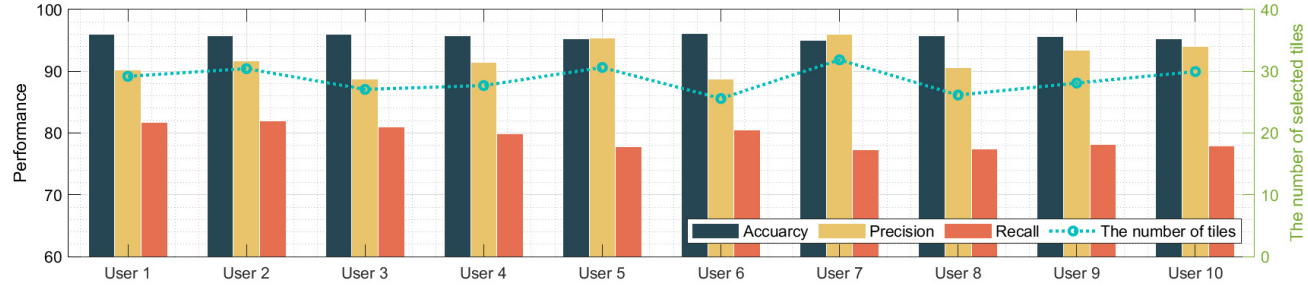


Fig. 9. Overall performance (accuracy, precision, recall, and the number of selected tiles) of different viewers for *Conan 1* with *FedLive*.

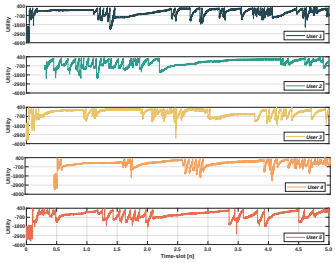


Fig. 10. The value variation of utility vs. time slot for different users.

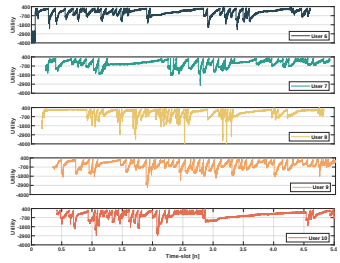


Fig. 11. The value variation of utility vs. time slot for different users.

solution for 10 different viewers (users 6-15 in the dataset [34]) for the video *Conan 1*. As Fig. 9 shows, there are small gaps in the prediction performance for different users when employing our solution, indicating that the prediction performance of *FedLive* is stable between different viewers. The above results show that our method can provide no less than 90% accuracy, around 90% precision, and about 80% recall for each viewer. Fig. 10 and Fig. 11 provide the value variation of utility for the 10 different viewers over time. The value of the utility function is calculated based on our objective function (10). The figures reveal that the utility varies between -1800 and -4000 in the exploration phase for users indexed 1 and 3-7. After the exploration phase, the viewers enter the exploitation phase, where the utility value remains above -1800 most of the time for all viewers. It is worth noting that some users do not go through the exploration phase since they, as subsequent viewers (users indexed 2 and 8-10), can directly download the FoV prediction model trained by previous viewers' clusters.

To verify the feasibility of our solution, we also test the **processing time** of the model training and FoV prediction

TABLE III
AVERAGE PROCESSING TIME(IN SECONDS) OF THE MODEL TRAINING AND FOV PREDICTION

Video Name		Conan	Skiing	Alien	War	Rhinos
Average Training Time (Epoch)	100	1.488	1.432	1.309	1.267	1.401
	200	2.942	2.843	2.598	2.501	2.798
	500	7.350	7.032	6.512	6.255	6.978
	1000	14.199	13.953	12.990	12.067	13.545
Prediction Time		0.258	0.247	0.235	0.230	0.246

modules in *FedLive*. The processing time of our model trained in different epochs for different videos is shown in Table III. As we can see, the training time increases with the epoch, and our solution can complete more than 100 iterations in 2 seconds. This indicates that each preceding viewer can train the model more than 100 iterations in advance and then gives the trained model to subsequent viewers who are behind a video segment (2s). The processing time of FoV prediction is short for all five videos, only about 0.2s, indicating that the model can be used for the FoV prediction of live 360-degree video streaming services.

To compare the service performance of the four solutions, we evaluated our solution in terms of four metrics:

- **Average resolution of FoV Tiles [Mbps]**: the average bitrate of tile within the viewer's FoV.
- **Bandwidth Saving [Mbps]:** in our system, the bandwidth is reduced by transmitting the tiles within FoV at a high bitrate and those outside FoV at a low bitrate. Thus, this metric is calculated as the bandwidth occupied by all tiles transmitting at high bitrates minus the bandwidth occupied by tiles within FoV transmitting at high bitrates and outside FoV transmitting at low bitrates.
- **Bitrate Change Ratio (BCR) [%]:** we define a video segment that has a different bitrate than the previous segment as a bitrate change. Then the BCR is the ratio of the number of video segments with a bitrate change to the total number of video segments.
- **Average Buffer Size [s]:** average duration of video content in the client's buffer.

Fig. 12 compares the solutions in terms of the four metrics. *FedLive* maintains the highest average resolution of selected tiles and the lowest bitrate change ratio, and is competitive on the performance of bandwidth saving and averaged buffer size compared with the other three solutions. Our approach outperforms *LiveDeep* and *LiveObj* in terms of service performance and has comparable performance compared with the offline solution *PanoSalNet*. To prove the feasibility of our solution, we compared the signalling cost of *FedLive* with *LiveDeep*, *LiveObj* and *PanoSalNet* as shown in Fig. 13. In order to facilitate the analysis, we assumed that the prediction models of the four solutions have equal sizes. Although, *FedLive* requires a higher number of message exchanges than other solutions, the two-layer delivery structure of the method restricts the signaling costs, which increases linearly with the number of viewers and is therefore scalable.

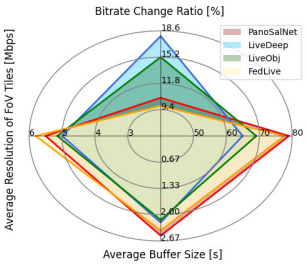


Fig. 12. The results of service performance and bandwidth efficiency

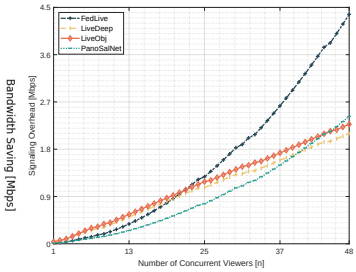


Fig. 13. Signalling overhead of control message for four solutions.

VII. CONCLUSIONS

This paper proposed *FedLive*, a novel federated transmission framework for PLS, which jointly optimizes the FoV prediction and content delivery. First, we designed a gradient-based viewer clustering method to integrate resources of the viewers with similar viewing behavior that also enables cooperative learning for the prediction model. We converted the FoV prediction to an online inference problem and formulated it in the RVI framework. Then, we transformed the inference problem into a joint optimization of prediction and delivery by reformulating the object function and redefining the 4-tuple Markov decision process. With the inspiration of clustered federated learning, a clustered federated updating is designed to facilitate training of the RVI-structure-based model asynchronously. In addition, we devised two practical algorithms, GVC and PAD, to achieve viewer clustering and joint optimization of delivery and learning, respectively. Based on the public dataset [37], we carried out a series of data-driven experiments and compared the performance of our solution with a synchronized benchmark and three state-of-the-art solutions [10], [14], [15]. The results show that our solution outperforms the alternative solutions in terms of prediction accuracy, bandwidth saving, and latency. In future work, more deep learning algorithms, such as soft actor-critic [55] and graph learning [56], will be studied in our federated transmission framework to make the solution more comprehensive.

ACKNOWLEDGMENT

This work was supported in part by the Fundamental Research Funds for the Central Universities and by the National Science Fund for Distinguished Young Scholars under grant 62225105. This work was supported in part by the National Natural Science Foundation of China under Grant No. 61906159, and in part by the Natural Science Foundation of Sichuan under Grant 23NSFSC5094, 23NSFSC0114. G.-M. Muntean acknowledges the Science Foundation Ireland Research Centres Programme support via grant 12/RC/2289_P2 (Insight).

REFERENCES

[1] H. Duan, J. Li, S. Fan, Z. Lin, X. Wu, and W. Cai, *Metaverse for Social Good: A University Campus Prototype*. New York, NY, USA: Association for Computing Machinery, 2021, p. 153–161.
[2] H. iLab. (2017) Cloud vr solution white paper. [Online]. Available: https://www-file.huawei.com/-/media/corporate/pdf/ilab

[3] H. K. Kim, J. Park, Y. Choi, and M. Choe, "Virtual reality sickness questionnaire (vrsq): Motion sickness measurement index in a virtual reality environment," *Applied ergonomics*, vol. 69, pp. 66–73, 2018.

[4] Cisco, "Cisco annual internet report (2018–2023) white paper," *https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report*, 2018.

[5] A. Yaqoob and G.-M. Muntean, "A combined field-of-view prediction-assisted viewport adaptive delivery scheme for 360° videos," *IEEE Transactions on Broadcasting*, vol. 67, no. 3, pp. 746–760, 2021.

[6] X. Hou, S. Dey, J. Zhang, and M. Budagavi, "Predictive adaptive streaming to enable mobile 360-degree and vr experiences," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[7] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Fixation prediction for 360° video streaming in head-mounted virtual reality," in *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2017, p. 67–72.

[8] A. Nguyen, Z. Yan, and K. Nahrstedt, "Your attention is unique: Detecting 360-degree video saliency in head-mounted display for head movement prediction," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18, 2018, pp. 1190–1198.

[9] M. Qiao, M. Xu, and et al., "Viewport-dependent saliency prediction in 360° video," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[10] C. Wu, R. Zhang, Z. Wang, and L. Sun, "A spherical convolution approach for learning long term viewport prediction in 360 immersive video," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 14 003–14 040.

[11] J. Chen, X. Luo, M. Hu, D. Wu, and Y. Zhou, "Sparkle: User-aware viewport prediction in 360-degree video streaming," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[12] M. F. R. Rondón, L. Sassatelli, R. Aparicio-Pardo, and F. Precioso, "A unified evaluation framework for head motion prediction methods in 360° videos," in *Proceedings of the 11th ACM Multimedia Systems Conference*, ser. MMSys '20, 2020, p. 279–284.

[13] O. Eltobgy, O. Arafa, and M. Hefeeda, "Mobile streaming of live 360-degree videos," *IEEE Transactions on Multimedia*, vol. 22, no. 12, pp. 3139–3152, 2020.

[14] X. Feng, Y. Liu, and S. Wei, "Livedeep: Online viewport prediction for live virtual reality streaming using lifelong deep learning," in *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, ser. VR '20, 2020, pp. 800–808.

[15] X. Feng and et al., "Liveobj: Object semantics-based viewport prediction for live mobile virtual reality streaming," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 5, pp. 2736–2745, 2021.

[16] Z. Jiang, X. Zhang, Y. Xu, Z. Ma, J. Sun, and Y. Zhang, "Reinforcement learning based rate adaptation for 360-degree video streaming," *IEEE Transactions on Broadcasting*, pp. 1–15, 2020.

[17] H. Pang, C. Zhang, F. Wang, J. Liu, and L. Sun, "Towards low latency multi-viewpoint 360° interactive video: A multimodal deep reinforcement learning approach," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, ser. INFOCOM '19, 2019, pp. 991–999.

[18] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "Drl360: 360-degree video streaming with deep reinforcement learning," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, ser. INFOCOM '19, 2019, pp. 1252–1260.

[19] S. Z. Yang, J. Hu, K. Jiang, H. Xiao, and M. Wang, "Hybrid-360: An adaptive bitrate algorithm for tile-based 360 video streaming," *Transactions on Emerging Telecommunications Technologies*, vol. 33, 2022.

[20] Y. Ban, Y. Zhang, H. Zhang, X. Zhang, and Z. Guo, "Ma360: Multi-agent deep reinforcement learning based live 360-degree video streaming on edge," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*, ser. ICME '20, 2020, pp. 1–6.

[21] X. Corbillon, F. De Simone, and G. Simon, "360-degree video head movement dataset," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, ser. MMSys'17, 2017, p. 199–204.

[22] L. Xie, X. Zhang, and Z. Guo, "Cls: A cross-user learning based system for improving qoe in 360-degree video adaptive streaming," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18, 2018, p. 564–572.

[23] Y. Bao, T. Zhang, A. Pande, H. Wu, and X. Liu, "Motion-prediction-based multicast for 360-degree video transmissions," in *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, ser. SECON '17, 2017, pp. 1–9.

[24] J. Chen, X. Qin, G. Zhu, B. Ji, and B. Li, "Motion-prediction-based wireless scheduling for multi-user panoramic video streaming," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, ser. INFOCOM '21, 2021, pp. 1–1.

[25] M. Zink, R. Sitaraman, and K. Nahrstedt, "Scalable 360° video stream delivery: Challenges, solutions, and opportunities," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 639–650, 2019.

[26] T. Biatek, W. Hamidouche, P. Cabarat, J. Travers, and O. Déforges, "Scalable video coding for backward-compatible 360° video delivery over broadcast networks," *IEEE Transactions on Broadcasting*, vol. 66, no. 2, pp. 322–332, 2020.

[27] R. Ghaznavi-Youvalari, A. Zare, A. Aminlou, M. M. Hannuksela, and M. Gabbouj, "Shared coded picture technique for tile-based viewport-adaptive streaming of omnidirectional video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 10, pp. 3106–3120, 2019.

[28] P. Maniotis, E. Bourtsoulatze, and N. Thomos, "Tile-based joint caching and delivery of 360° videos in heterogeneous networks," *IEEE Transactions on Multimedia*, vol. 22, no. 9, pp. 2382–2395, 2020.

[29] M. Dasari, A. Bhattacharya, S. Vargas, P. Sahu, A. Balasubramanian, and S. R. Das, "Streaming 360-degree videos using super-resolution," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, ser. INFOCOM '20, 2020, pp. 1977–1986.

[30] J. Kim, Y. Jung, H. Yeo, J. Ye, and D. Han, "Neural-enhanced live streaming: Improving live video ingest via online learning," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '20, 2020, p. 107–125.

[31] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2020.

[32] T. Weber, N. Heess, A. Eslami, J. Schulman, D. Wingate, and D. Silver, "Reinforced variational inference," in *Advances in Neural Information Processing Systems (NIPS) Workshops*, 2015.

[33] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[34] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in vr spherical video streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, ser. MMSys'17, 2017, p. 193–198.

[35] A. Yaqoob, T. Bi, and G.-M. Muntean, "A survey on adaptive 360° video streaming: Solutions, challenges and opportunities," *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2801–2838, 2020.

[36] G. He, J. Hu, H. Jiang, and Y. Li, "Scalable video coding based on user's view for real-time virtual reality applications," *IEEE Communications Letters*, vol. 22, no. 1, pp. 25–28, 2018.

[37] A. Nguyen and Z. Yan, "A saliency dataset for 360-degree videos," in *Proceedings of the 10th ACM Multimedia Systems Conference*, ser. MMSys '19, 2019, p. 279–284.

[38] E. Cui, D. Yang, H. Wang, and W. Zhang, "Learning-based deep neural network inference task offloading in multi-device and multi-server collaborative edge computing," *Transactions on Emerging Telecommunications Technologies*, vol. 33, 2022.

[39] R. Verma and S. Chandra, "Fogbus3: A scalable and reliable framework for integrated iot and fog computing scenario," *Transactions on Emerging Telecommunications Technologies*, vol. 33, 2022.

[40] A. Yaqoob, T. Bi, and G. M. Muntean, "A survey on adaptive 360° video streaming: Solutions, challenges and opportunities," *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2801–2838, 2020.

[41] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proceedings of the 25th ACM International Conference on Multimedia*, New York, NY, USA, 2017, p. 315–323.

[42] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using scalable video coding," in *Proceedings of the 25th ACM International Conference on Multimedia*, New York, NY, USA, 2017, p. 1689–1697.

[43] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in

Proceedings of the 25th ACM International Conference on Multimedia, New York, NY, USA, 2017, p. 315–323.

[44] X. Hou, S. Dey, J. Zhang, and M. Budagavi, "Predictive adaptive streaming to enable mobile 360-degree and vr experiences," *IEEE Transactions on Multimedia*, vol. 23, pp. 716–731, 2021.

[45] Y. Mao, L. Sun, Y. Liu, and Y. Wang, "Low-latency fov-adaptive coding and streaming for interactive 360° video streaming," New York, NY, USA, 2020, p. 3696–3704.

[46] L. Zhong, X. Chen, C. Xu, Y. Ma, M. Wang, Y. Zhao, and G.-M. Muntean, "A multi-user cost-efficient crowd-assisted vr content delivery solution in 5g-and-beyond heterogeneous networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2022.

[47] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[48] X. Chen, C. Xu, M. Wang, Z. Wu, S. Yang, L. Zhong, and G.-M. Muntean, "A universal transcoding and transmission method for livecast with networked multi-agent reinforcement learning," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[49] X. Chen, C. Xu, M. Wang, Z. Wu, L. Zhong, and L. A. Grieco, "Augmented queue-based transmission and transcoding optimization for livecast services based on cloud-edge-crowd integration," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 11, pp. 4470–4484, 2021.

[50] K. Zhang, Z. Yang, and T. Basar, "Networked multi-agent reinforcement learning in continuous spaces," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 2771–2776.

[51] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

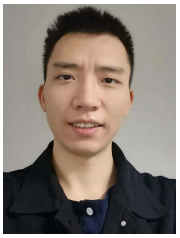
[52] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.

[53] M. Wang, C. Xu, X. Chen, H. Hao, L. Zhong, and D. O. Wu, "Design of multipath transmission control for information-centric internet of things: A distributed stochastic optimization framework," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9475–9488, 2019.

[54] H. Pang, C. Zhang, F. Wang, H. Hu, Z. Wang, J. Liu, and L. Sun, "Optimizing personalized interaction experience in crowd-interactive livecast: A cloud-edge approach," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1217–1225. [Online]. Available: https://doi.org/10.1145/3240508.3240642

[55] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[56] J. Tan, Q. Li, J. Wang, and J. Chen, "Finhgnn: A conditional heterogeneous graph learning to address relational attributes for stock predictions," *Information Sciences*, vol. 618, pp. 317–335, 2022.



Xingyan Chen received the Ph. D degree in computer technology from Beijing University of Posts and Telecommunications (BUPT), in 2021. He is currently a lecturer with the School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu. He has published papers in well-archived international journals and proceedings, such as the IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and IEEE INFOCOM etc. His research interests include Multimedia Communications, Multi-agent Reinforcement Learning and Stochastic Optimization.

Mu Wang received his M.S. and Ph.D. degrees in computer technology from Beijing University of Posts and Telecommunications (BUPT) in 2015 and 2020. He was a Joint Ph.D. student at the School of Electrical, Computer, and Energy Engineering (ECEE), Arizona State University. He is currently a postdoctoral research associate with the Department of Computer Science and Technology & BNRist, Tsinghua University. His research interests include information-centric networking, wireless communications, and multimedia delivery in wireless networks.

Changqiao Xu [SM'15] received the Ph.D. degree from the Institute of Software, Chinese Academy of Sciences (ISCAS) in Jan. 2009. He was an Assistant Research Fellow and R&D Project Manager in ISCAS from 2002 to 2007. He was a researcher at Athlone Institute of Technology and joint training PhD at Dublin City University, Ireland during 2007-2009. He joined Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in Dec. 2009. Currently, he is a Professor with the State Key Laboratory of Networking and Switching Technology, and Director of the Network Architecture Research Center at BUPT. His research interests include Future Internet Technology, Mobile Networking, Multimedia Communications, and Network Security. He has edited two books and published over 200 papers in prestigious international journals and conferences, including IEEE/ACM ToN, IEEE TMC, IEEE INFOCOM, ACM Multimedia etc. He has served a number of international conferences and workshops as a Co-Chair and TPC member. He is currently serving as the Editor-in-Chief of TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES (WILEY). He is a Senior member of IEEE.

Yu Zhao received the B.S. degree from Southwest Jiaotong University in 2006, and the M.S. and Ph.D. degrees from the Beijing University of Posts and Telecommunications in 2011 and 2017, respectively. He is currently an Associate Professor at Southwestern University of Finance and Economics. His current research interests include natural language processing, knowledge graph, machine learning, and recommendation system.

Ke Jiang received the B.S. degree from Beijing University of Posts and Telecommunications in 2022. She is currently a master student at Beijing University of Posts and Telecommunications. Her current research interests include multimedia communication and reinforcement learning.

Yang Shujie received the Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2017. He is an Lecture with Beijing University of Posts and Telecommunications. His current research interests include the areas of VR networks, content delivery network, and wireless networking.

Qing Li received his PhD degree from Kumoh National Institute of Technology in February of 2005, Korea, and his M.S. and B.S. degrees from Harbin Engineering University, China. He is a postdoctoral researcher at Arizona State University and the Information & Communications University of Korea. He is a professor at Southwestern University of Finance and Economics, China. His research interests include natural language processing, FinTech. He has published more than 70 papers in the prestigious refereed conferences and journals, such as IEEE TKDE, ACM TOIS, AAAI, SIGIR, ACL, WWW, etc.

Lujie Zhong received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2013. She is currently an Associate Professor with the Information Engineering College, Capital Normal University, Beijing, China. She has published papers in prestigious international journals and conferences, including IEEE COMMUNICATION MAGAZINE, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE INTERNET THINGS JOURNAL, IEEE INFOCOM and ACM MULTIMEDIA, etc. Her research interests include communication networks, computer system and architecture, and mobile Internet technology.

Gabriel-Miro Muntean [F 22] is a Professor with the School of Electronic Engineering, Dublin City University (DCU), Ireland, and co-Director of DCU Performance Engineering Laboratory. He has published 4 books and over 450 papers in top international journals and conferences. His research interests include rich media delivery quality, performance, and energy-related issues, technology enhanced learning, and other data communications in heterogeneous networks. He is an Associate Editor of the IEEE TRANSACTIONS ON BROADCASTING, the Multimedia Communications Area Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and reviewer for important international journals, conferences, and funding agencies. He coordinated the EU project NEWTON and leads the DCU team in the EU project TRACTION.