This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

Buffer-Aware Virtual Reality Video Streaming with Personalized and Private Viewport Prediction

Ran Zhang*‡, Jiang Liu*‡, Fangqi Liu*, Tao Huang*‡, Qinqin Tang*, Shangguang Wang*, and F. Richard Yu†, *Fellow, IEEE*

*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China, 100876

‡Purple Mountain Laboratories, Nanjing, China, 211111

†Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, K1S 5B6, Canada

***Abstract*—Viewport prediction and prefetch have an important influence on VR video streaming performance. This work proposes a novel federated learning-based viewport prediction model training algorithm, ComPer-FedAvg. The proposed algorithm leverages a VR video's common viewing pattern and users' personal viewing patterns to train the prediction model in a distributed and privacy-preserving manner. Further, considering the VR video viewport prediction accuracy, a stochastic game is formulated to solve the VR streaming network's communication resource allocation problem, where limited communication resource blocks are auctioned to users to achieve the optimal overall VR viewing experience. For each user, the auction is decomposed into two disjoint subproblems, namely, the optimal number of data rate requesting and true value claiming (bidding). The optimal true value claiming has been analytically proved to be equal to the VR viewing reward with given data rate. Due to the lack of global information when users request data rate, we reformulate users' data rate requesting problem as a POMDP problem. A novel deep reinforcement learning algorithm is adopted to solve the problem. Evaluation and simulation results show the proposed viewport prediction and VR streaming schemes outperform conventional solutions in terms of prediction accuracy and VR viewing experience.**

***Index Terms*—Virtual Reality, Personalized Federated Learning, Deep Reinforcement Learning, Communication Resource Allocation.**

I. Introduction

With the development of information and communication theory and technology, many novel applications have emerged, which promise disruptive changes to people's lives and vast markets. Virtual Reality (VR) video streaming is one of the most promising applications, which provides an unprecedented virtual and immersive experience for users. However, numerous technical challenges stemming from the QoE and efficiency requirements need to be addressed before its success. More specifically, compared to conventional video streaming, VR video provides high-resolution 360° visual fields on three Degrees of Freedom (3-DoF). The broader and higher-definition visual field means more pixels to transmit, which requires a much higher bandwidth. Besides, as users move their heads, corresponding viewports need to be presented to users within an ultra-low latency; otherwise, users will suffer dizziness and nausea [1].

Jiang Liu is the corresponding author: liujiang@bupt.edu.cn

Tiling and prefetch are among the most important solutions to improve VR video streaming efficiency. Compared to streaming the entire 3-DoF VR video, tiling schemes split the VR video into discrete tiles in the spatial dimension. VR video providers only stream tiles currently viewed by users, where fewer tiles to transmit means lower bandwidth consumption. The other solution prefetch segments the VR video in the temporal dimension. If a user's data rate is sufficient for the currently viewed segment, it can spare the current data rate to download future tiles into the local buffer. The prefetch scheme can reduce the motion response latency and stabilize the VR streaming performance in the dynamic communication environment.

Although the tiling and prefetch schemes promise a considerable improvement, their performance depends heavily on the viewport prediction accuracy. Since prefetch is executed on the granularity of tile, the prefetched tiles can only be of improvement when the user finally watches them. If the prediction accuracy is low, a large proportion of the data rate will be consumed to download tiles that will not be viewed, which significantly degrades the system performance. Meanwhile, state-of-the-art viewport prediction solutions [2]–[6] usually take advantage of Deep Neural Networks (DNN), which requires a large amount of data to train the prediction model. However, due to the rising privacy concern, users are unwilling to share their viewing history with others for prediction model training, and VR video providers must not share their users' viewing history with others without permission. The obstacles above limit the data efficiency of the prediction model training. Moreover, the viewing patterns among different users and VR videos are different [7], so the prediction model needs to consider both personal and common viewing patterns, where limited works have been done. Besides, under a given prediction performance, how to tradeoff between communication resource consumption and video prefetch among multiple VR video viewers remains an open issue.

Recently, the emerging Mobile Edge Computing (MEC) [8], Federated Learning (FL) [9], and Deep Reinforcement Learning (DRL) [10] bring new promise to solve the challenges above. The MEC scheme provides caching, computing, and communication service to users in an integrated manner, enabling upper-layer applications to sense underlying

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

resource information and achieve cross-layer optimization. Meanwhile, FL can achieve cross-data-owner model training without exposing their personal data, which provides a privacy-preserving scheme to train the model cooperatively. DRL can learn to solve long-term dynamic problems in a model-free manner, and it scales well for the problem with the curse of dimension, whose improvement has been widely validated [10]–[12].

In this work, inspired by recent efforts, we propose an MEC-enabled VR streaming system, where VR video viewport prediction and communication resource allocation are integrated to achieve efficient VR streaming. Considering the video-common and user-personalized viewing patterns, we propose a novel Common-Personalized Federated Averaging (ComPer-FedAvg) algorithm to learn the VR video viewing pattern in a distributed and privacy-preserving fashion. Meanwhile, to maximize the overall VR video watching experience, we adopt the Vickrey–Clarke–Groves (VCG) [12] auction to decide the communication resource block allocation. Further, we formulate the communication resource allocation problem into a stochastic game, where users consider their prefetch buffer and channel state information to request communication data rate. Due to the lack of global state information in the resource competition process, we reformulate the problem into a Partially Observable Markov Decision Process (POMDP) for each user. Finally, to solve the problem, we adopt a DRL algorithm to learn the system model and make decisions in the dynamic environment. The contributions of this work are summarized as follows.

- We propose an MEC-enabled VR streaming system, where VR viewport prediction and communication resource allocation are integrated to achieve efficient VR streaming.
- We propose ComPer-FedAvg, a novel FL-based prediction model training algorithm. In a distributed and privacy-preserving manner, ComPer-FedAvg can utilize the viewing history in both users' devices and centralized VR video providers to train each user's personalized prediction model for each VR video.
- We formulate the multi-user buffer-aware VR video streaming problem as a stochastic game, where users bid for communication data rate to maximize their VR video viewing utility. Considering the lack of global state information in the resource competition process, the problem is further reformulated into a POMDP problem for each VR user.
- We leverage a DRL algorithm to solve the formulated resource block allocation problem in a distributed manner. Evaluation results on real data traces validate the improvement of the proposed prediction training algorithm and VR video streaming performance.

The rest of the work is organized as follows. In section II, we present related works on VR video viewport prediction and VR video streaming. In section III, we present the system model of the proposed VR video streaming system. We describe the proposed ComPer-FedAvg algorithm in section IV. Then in section V, we formulate the VR video streaming problem into a stochastic game and reformulate it into a POMDP problem. To solve the formulated problem, we decompose the original problem and present the DRL algorithm in section VI. Evaluations and discussions are conducted in section VII. In section VIII, we conclude the work.

II. Related Works

In this section, we present related works on VR viewport prediction and VR video streaming.

A. VR Video Viewport Prediction

The efforts to predict viewport can be classified into two categories, namely, the single user based prediction and multiple users based prediction.

The single user based prediction efforts focus on the insight of a single viewer's viewing pattern to make predictions. In [13], the author conducted linear regression on users' viewing history, and they predicted future viewports based on current viewing behavior. Also, considering the decreasing accuracy of prediction as the prediction windows increases, they accumulated the prediction results as the final prediction output. There are also efforts that concentrated on the content of the VR video to make predictions. In [2], the author proposed to extract visual features of the viewport from different dimensions, based on which saliency map is predicted with a Convolutional Neural Network (CNN). Then with the information of the predicted saliency map, the authors predicted the viewport based on the proposed concept of visual equilibrium and uncertainty. Meanwhile, the authors of [3] proposed to track the moving and static objects in the user's viewport and transit tiles containing the currently watched objects; besides, they adjusted the tile transmission decision according to the performance of tracking and transmission accuracy. A similar idea to object tracking is the viewport tracking filter proposed in [4], where the authors adapted an object tracking algorithm to a viewport tracking filter, and they balanced between the tracking system and a Recurrent Neural Network (RNN) to output the final prediction results. Also considering predicting a series of viewports in the future, the author of [5] extracted all features of VR content that has not been viewed, and input the extracted features as well as head movement to the Long Short Term Memory (LSTM) network for a series of viewports prediction in the future.

The multiple users based prediction efforts focus on exploiting the common feature among different users. In [14], the author conducted linear regression on history head movement, and when predicting a new viewport based on the trained model, the author adjusted the results with K-Nearest-Neighbor users' viewport to correct the prediction bias. A similar idea was also presented in [7], where the prediction was conducted on the horizontal and vertical axis, and the final output is a weighted sum of the user's prediction and other users' prediction. Clustering is another approach to integrate the viewing patterns of different users. In [15] and [16], the authors clustered the viewers into different clusters according to their viewing habit, and they calculated a common viewing pattern for a cluster and hence made

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

predictions for users in the corresponding cluster. In [6], the authors adopted transfer learning for the saliency map prediction, which shared the prediction model among different users. Then the predicted saliency map together with the current head orientation was input to the LSTM for the next viewport prediction.

Remark: As can be seen, the trend for viewport prediction is to incorporate both VR content features and head movement. Meanwhile, the utilization of multiple users' knowledge and experience can enhance the resilience and accuracy of the single user based prediction approaches. However, as can be seen, the privacy-preserving prediction cooperation has been largely neglected.

B. VR Video Streaming

The efforts to improve VR video delivery can be classified into two categories, i.e., flexible video delivery and infrastructure assistance.

For flexible video delivery, the authors of [17] proposed to split the visual fields into tiles and only transmit the tiles within users' visual fields. In [18], the authors further improved the flexibility. Compared to only deliver currently viewed visual fields, they reconstructed the spherical visual fields and streamed the viewed visual fields in priority. Another solution [19] utilized the knowledge of ROI, where they streamed the ROI regions of the video with high quality while compromising the quality of rest regions. In [**?**], the authors further proposed a series of new factors to consider when evaluating the VR video quality, such as head movement speed, brightness, contrast, and depth variation. Based on the new concerns, the author introduced adaptive tiling and quality to improve the real-time streaming performance.

For infrastructure assistance, the authors of [1] proposed to integrate mmWave and MEC to achieve low-latency VR video delivery. They pre-computed the frames that users might request, cache the computing result locally, and send the results to users via mmWave communication channels. The authors of [20] considered the VR video tracking accuracy and video processing and transmitting latency, and they optimized VR video delivery under the cooperation of multiple small base stations. In [21], the authors further considered the correlation between different users' viewing videos, and they utilized the information to decide uplink and downlink resource allocation. In [11], the authors made use of a federated learning approach to learn the gesture of users, and thus retrieve the wireless channel blockage predictions. Based on the predicted results, the authors optimized mmWave base station association and achieved the minimal break of presence. In [22], the authors considered ROI and field of view of VR videos, predicted the viewing orientation of users, and cached low-resolution video at every node. Integrating the knowledge above, the authors streamed VR video with multicast. Compared to offload tasks to MEC servers, the authors of [23] proposed to offload the VR rendering tasks from MEC nodes to users and achieve minimal bandwidth requirement. The author of [24] considered the projection of VR videos, and they formulated the projection version caching and computing offloading into a multiple-choice multiple dimensional knapsack problem. Besides cellular connections, the authors of [25] further integrated Wi-Fi into consideration, where they divided the VR video into low-quality base layers and high-quality enhancement layers. They delivered encoded base layers to users and decoded enhancement layers to users for ultra-low latency VR video delivery.

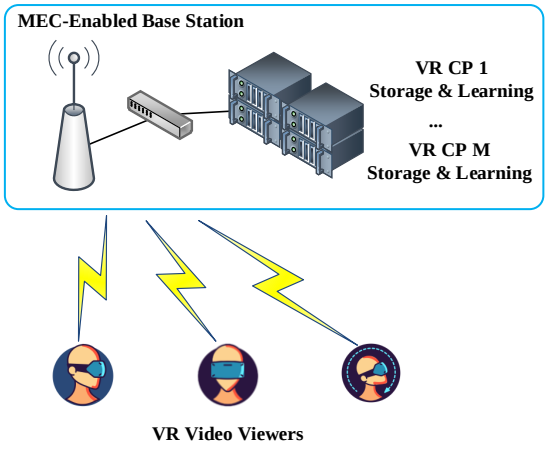


Fig. 1: Overall system architecture.

Remark: The progress of MEC has enabled cross-layer optimization, which incorporates the underlying caching, computing, and communication resources to enhance the VR streaming performance. However, although the excellent efforts have been made above, it has been largely neglected that the VR video prefetch influences the viewing experience, which is related to the prediction accuracy. Further, in the process of underlying resource allocation, prefetch based buffer conditions should be considered for higher efficiency and better service quality.

III. System Design and Model

In this section, we present the overall system design and the model on viewport prediction, VR video watching, and wireless communication resource allocation.

A. System Design

The overall system architecture is presented in Fig. 1. As is shown in the figure, there are N users connecting to the MEC-enabled base station, and there are C pieces VR videos provided by M VR video content providers (CPs). The VR videos are organized in spatial and temporal dimensions as in Fig. 2. More specifically, VR videos are streamed to users as segments, each containing T seconds of video to be watched. Meanwhile, for each segment, VR videos are split into tiles for flexible transmission and data rate saving. Denote $\boldsymbol{V}_t \in \{0,1\}^{N\times C\times L}$ as all users' viewport in the t-th segment, where L is the total number of tiles for each segment, and $V_{tikl} = 1$ means the tile l of VR video k is watched by the user i at the t-th segment. Moreover, VR videos are encoded into high and low-quality versions. VR video users can download high or low-quality tiles according

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

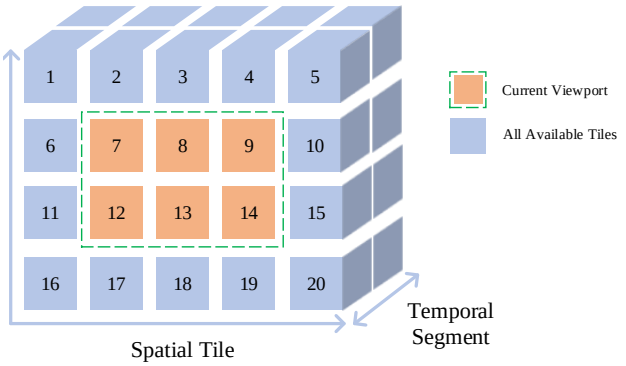


Fig. 2: VR video organization.

to their hardware profiles, such as computing power and available data rate. Without losing generality, we define the bitrate of different tiles as the same value, and the bitrate of a single high and low-quality tile is denoted as s^H and s^L, respectively.

Users watch the video in temporally sequential order. When watching a segment, users predict the tiles to be watched in the next segment according to a prediction model trained in advance. If they have spare data rate, they can also prefetch tiles in the next segment as they watch the current segment.

B. Prediction Model Training Model

The prediction model predicts each VR users' next viewport based on the current viewing state. More specifically, each user adopts the training model to predict tiles viewed during the $t+1$-th segment based on the tiles viewed in the t-th segment. In the following, we present the model of training the personalized prediction model. Note that, in this work, we focus on the personalized model training algorithm rather than the specific prediction algorithm; therefore, we take tile-based prediction as an example.

The prediction model is trained based on users' viewing history. Due to the privacy concern, each user maintains their own unique viewing history data set denoted by $\boldsymbol{\mathcal{D}}_i^u = \cup_{1\leqslant k\leqslant C}\boldsymbol{\mathcal{D}}_{ik}^u$, where $\boldsymbol{\mathcal{D}}_{ik}^u$ is the user i's viewing history on VR video k. Meanwhile, $\boldsymbol{\mathcal{D}}_{ik}^u$ is organized as $\boldsymbol{\mathcal{D}}_{ik}^u = \{\boldsymbol{V}_{tik}|t = 1,2,3,\cdots\}$, where $\boldsymbol{V}_{tik} = [V_{tik1}, V_{tik2}, \cdots, V_{tikL}]$. Apart from viewing history maintained by users, CPs can also collect users' viewing history on a specific video they provide. The viewing history maintained by CPs is denoted as $\boldsymbol{\mathcal{D}}_{jk}^c = \{V_{tikl}|t \geqslant 1, 1 \leqslant i \leqslant N, 1 \leqslant l \leqslant L\}$, which represents the CP j's collected viewing history on video k.

Based on the user and CP-maintained data set, prediction models are trained for users to predict the next viewport. Considering the distributed data set and different user viewing patterns, the prediction model training algorithm adopts a personalized federated learning-based framework [26]. In the personalized learning-based framework, there is a centralized learning node and a number of distributed learning nodes. All learning nodes maintain the same prediction model architecture but with different weight parameters $\boldsymbol{\omega}$. The central node steers the personalized federated learning process, and its weight is denoted as the common weight $\boldsymbol{\omega}_k^c$. The central node updates $\boldsymbol{\omega}_k^c$ based on the distributed data set $\boldsymbol{\mathcal{D}}^u$ and $\boldsymbol{\mathcal{D}}^c$. Before users start to view the VR video k, they download the prediction model $\boldsymbol{\omega}_k^c$ from the central learning node, and for model personalization, they update $\boldsymbol{\omega}_k^c$ to get their personalized $\boldsymbol{\omega}_{ik}^u$ based on their own local data set $\boldsymbol{\mathcal{D}}_{ik}^u$ as in [26], [27]. Due to limited computing capacity equipped in user devices, the prediction model is updated by several mini-batch gradient descent [28] on the local data set.

More specifically, the VR video users are the distributed learning nodes. A user i maintains a consistent neural network $\mathcal{N}_{ik}$ with personalized $\boldsymbol{\omega}_{ik}^u$ for a VR video k. Given a group of consecutive input viewports $[\boldsymbol{V}_{(t-n)ik}, \cdots, \boldsymbol{V}_{(t-1)ik}, \boldsymbol{V}_{tik}]$, where $n+1$ is the prediction windows size, the neural network generates an estimation for the next viewport $\hat{\boldsymbol{V}}_{(t+1)ik} = \mathcal{N}_{ik}(\boldsymbol{\omega}_{ik}; [\boldsymbol{V}_{(t-n)ik}, \cdots, \boldsymbol{V}_{(t-1)ik}, \boldsymbol{V}_{tik}])$. For simplicity, in the following, we abbreviate $[\boldsymbol{V}_{(t-n)ik}, \cdots, \boldsymbol{V}_{(t-1)ik}, \boldsymbol{V}_{tik}]$ into $\boldsymbol{V}_{tik}$, and the prediction window is adjustable for different scenarios. The error for this prediction can be represented by $e_i(\boldsymbol{\omega}_{ik}; \boldsymbol{V}_{tik}, \boldsymbol{V}_{(t+1)ik})$, where $e_i(\cdot) : \mathbb{R}^L \to \mathbb{R}$ is the user i specified error measurement for supervised learning. Then the loss function $l_i(\cdot)$ for the user i's prediction model $\mathcal{N}_{ik}$ can be defined as (1), where p_i denotes the viewport distribution pattern of user i. Further, personalized $\boldsymbol{\omega}_{ik}^u$ can be derived by (2), where α^p is the update step size of the mini-batch gradient descent.

$$l_i(\boldsymbol{\omega}_{ik}) := \mathbb{E}_{(\boldsymbol{V}_{tik}, \boldsymbol{V}_{(t+1)ik})\sim p_i} e_i(\boldsymbol{\omega}_{ik}; \boldsymbol{V}_{tik}, \boldsymbol{V}_{(t+1)ik}) \quad (1)$$

$$\boldsymbol{\omega}_{ik}^u = \boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c) \quad (2)$$

Meanwhile, the CP that resides in the MEC server is the central learning node, and it sets the training objective and collects updates from distributed learning nodes. Denote by $\mathcal{L}(\boldsymbol{\omega}_k^c)$ the overall loss at the central node, then the training objective is to find the optimal common weight $\boldsymbol{\omega}_k^{c\star}$ that minimizes the central loss $\mathcal{L}(\boldsymbol{\omega}_k^c)$ as is shown in (3)

$$\boldsymbol{\omega}_k^{c\star} = \arg\min_{\boldsymbol{\omega}_k^c} \mathcal{L}(\boldsymbol{\omega}_k^c) \quad (3)$$

C. VR Video Watching Model

When watching the t-th segment of a VR video, users can prefetch tiles of the $t+1$-th segment. Also, when the user i begins to watch the t-th segment, it has a set of prefetched tiles stored in the local buffer, denoted as $\boldsymbol{B}_{tik}^H \in \{0,1\}^L$. $B_{tikl}^H = 1$ means the high-quality version of the tile l has been prefetched when the user i starts to watch the t-th segment of the VR video k. When all the tiles viewed in the t-th segment have been prefetched, i.e., $\boldsymbol{B}_{tik}^H \succeq \boldsymbol{V}_{tik}$, the user can directly watch the segment and obtain a satisfying experience. Otherwise, the user needs to download the missing tiles. The downloading decision is denoted as $\boldsymbol{\mathcal{X}}_{tik} \in \{0,1\}^L$, and $\mathcal{X}_{tikl} = 1$ means the user downloads the high-quality version of the tile l for the current segment. For the remaining missing tiles, the user downloads the low-quality version to make up a

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

TABLE I: Notations and Definitions

Notations	Definitions
T	The length of the VR video segment.
$\boldsymbol{V}_t$	Users' viewport for the t-th segment.
N	Number of end users.
C	Number of available VR videos.
M	Number of CPs.
L	Number of tiles in the complete 3-DoF horizon.
s^H, s^L	The bitrate of a high/low quality tile.
$\boldsymbol{\mathcal{D}^u}$, $\boldsymbol{\mathcal{D}^c}$	Viewing history data set of users and CPs.
$\mathcal{N}_{ik}$	User i's prediction model for VR video k.
$\boldsymbol{\omega}^u_{ik}$, $\boldsymbol{\omega}^c_k$	$\boldsymbol{\omega}^u_{ik}$ is the weight for the user i's personalized prediction model on VR video k. $\boldsymbol{\omega}^c_k$ is the common weight for the prediction model in the central node.
p_i	The viewport distribution pattern of user i.
b^p_{tji}	The data rate user i can achieve over the resource block j.
b_{ti}	User i's available data rate when start watching the t-th segment.
Φ^H_{tik}, Φ^L_{tik}	The upper and lower data rate bounds that affect users' viewing experience.
$\boldsymbol{B}^H_{tik}$, $\boldsymbol{B}^L_{tik}$	User i's prefetched high and low-quality tiles for the t-th segment of VR video k.
$\boldsymbol{\mathcal{X}}_{tik}$	User i's tile downloading decision for the t-th segment of VR video k.
r_{tik}	Instantaneous reward for user i watching the t-th segment of video k.
S	The number of resource blocks at the base station.
ξ_{ti}	The number of resource blocks the user i requests at the t-th scheduling slot.
$\hat{v}_{ti}$, v_{ti}	The claimed and real true value of the user i at the t-th scheduling slot.
$\boldsymbol{\psi}_t$	The resource block auction winner decision at the t-th scheduling slot.
$\boldsymbol{\phi}_t$	The resource block allocation at the t-th scheduling slot.
σ_{ti}	The payment the user i needs to make in the t-th scheduling slot.
α^p, α^t	The step size for the personalization and ComPer-FedAvg local update, respectively.
ι^u, ι^c	The weight balancing the trained model's preference over users' personal viewing pattern and the VR video's common viewing pattern.
$\boldsymbol{D}^u_i$, $\boldsymbol{D}^{u\prime}_i$, $\boldsymbol{D}^{u\prime\prime}_i$	Mini-batches of data samples from user i's viewing history.
$\boldsymbol{D}^c_{jk}$	Mini-batches of data samples of VR video k's viewing history stored in CP j.
$\boldsymbol{\mathcal{S}}_t$, $\boldsymbol{\mathcal{S}}_{ti}$	The global state of the user in the VR streaming system.
$\boldsymbol{\pi}$, π_i, π_{-i}	$\boldsymbol{\mathcal{S}}_t$ is the joint control policy of all users, π_i is the control policy of user i, and π_{-i} is the joint control policy of users expect i.
$\boldsymbol{\mathcal{A}}_t$, $\mathcal{A}_{ti}$	The action of all users and user i at the t-th scheduling slot.
$\mathcal{U}_{ti}$	The utility user i achieves on given state and action at the t-th scheduling slot.
$\mathcal{V}(\cdot)$	State value function of a given state and policy.
$Q(\cdot)$	Action value function of a given state and action.
$\boldsymbol{\mathcal{O}}_{ti}$	User i's observation at the t-th scheduling slot.
$\boldsymbol{\mathcal{M}}_i$	The experience memory of the user i.
$\mathcal{L}(\cdot)$, $\mathcal{L}^{D3RQN}(\cdot)$	$\mathcal{L}(\cdot)$ is the loss functions of ComPer-FedAvg algorithm, and $\mathcal{L}^{D3RQN}(\cdot)$ is the loss function of the D3RQN algorithm.

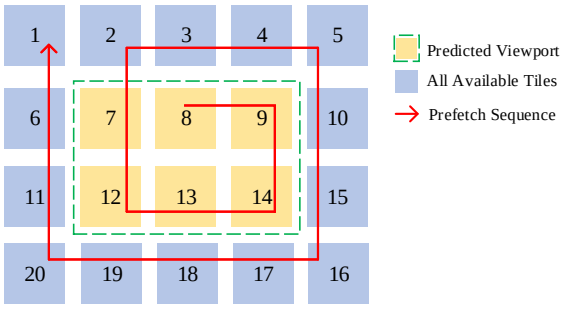


Fig. 3: VR video prefetch sequence.

complete viewport. Note that, as defined in (4) and (5), there is an upper data rate bound Φ^H_{tik} and a lower data rate bound Φ^L_{tik} that can affect the user's decision. In (4) and (5), $\circ$ is the Hadamard product operator.

$$\Phi^H_{tik} = ||\boldsymbol{V}_{tik} - (\boldsymbol{V}_{tik} \circ \boldsymbol{B}^H_{tik})||_1 \cdot s^H \tag{4}$$

$$\begin{aligned}\Phi^L_{tik} = ||(\boldsymbol{V}_{tik} - \boldsymbol{V}_{tik} \circ \boldsymbol{B}^H_{tik}) - \\ (\boldsymbol{V}_{tik} - \boldsymbol{V}_{tik} \circ \boldsymbol{B}^H_{tik}) \circ \boldsymbol{B}^L_{tik}||_1 \cdot s^L\end{aligned} \tag{5}$$

If the user i's available data rate is higher than the upper bound Φ^H_{tik}, i.e., the sum bitrate of high-quality missing tiles in the viewport, then the user can watch the high-quality VR video smoothly without stall. Meanwhile, if the user i's available data rate is lower than the lower bound Φ^L_{tik}, i.e., the sum bitrate of low-quality missing tiles, then the user cannot see a complete scene. When the user's available data rate is between Φ^L_{tik} and Φ^H_{tik}, it downloads a proportion of tiles in high quality, and the remaining is downloaded as low-quality ones. The objective of selecting high or low-quality tiles to download is to maximize the user's instantaneous VR video viewing reward as defined in (6).

$$r_{tik} = \begin{cases} 1, & b_{ti} \geqslant \Phi^H_{tik} \\ \frac{||\boldsymbol{B}^H_{tik} \circ \boldsymbol{V}_{tik} + (\boldsymbol{V}_{tik} - \boldsymbol{V}_{tik} \circ \boldsymbol{B}^H_{tik}) \circ \boldsymbol{\mathcal{X}}_{tik}||_1}{||\boldsymbol{V}_{tik}||_1}, & \Phi^L_{tik} \leqslant b_{ti} \leqslant \Phi^H_{tik} \\ -1, & b_{ti} \leqslant \Phi^L_{tik} \end{cases} \tag{6}$$

When the user can watch the current segment smoothly, it allocates the remaining data rate to prefetch the predicted tiles in the next segment. The prefetch rule is to first download the central tile in the predicted viewport and then proceed to the remaining tiles in clockwise order as shown in Fig. 3. The prefetch logic is consistent with that in [29], where the prefetch starts from the center of the predicted viewport, and an enlarged prefetch region promises a higher probability of overlapping with the actual viewport[1]. Since the prediction is not necessarily accurate, after downloading all high-quality tiles in the predicted viewport, the user proceeds to download tiles in the predicted viewport's outer rim.

[1]Other prefetch rules can also be supported, which does not influence the superiority of the proposed solution

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

D. VCG Auction-based Communication Model

The users are connected to the base station, and we assume Orthogonal Frequency Division Multiplexing (OFDM) is adopted. The base station has S resource blocks to allocate to users, and each resource block is a group of sub-carriers with an overall bandwidth of $\mathcal{B}$.

At the t-th scheduling slot, a resource block j promises a data rate b^p_{tji} available to user i, which is shown as

$$b^p_{tji} = \mathcal{B}\log_2(1 + \frac{P g_{tji} d_{ti}^{-\beta}}{\nu^2}) \tag{9}$$

where $\frac{P g_{ti} d_{ti}^{-\beta}}{\nu^2}$ is the Signal-to-Noise Ratio (SNR), P is the power of the base station, g_{tji} is the Rayleigh fading parameter of the resource block j for user i, d_{ti} is the distance from the base station to the user i, and β is the path loss exponent. In this work, we assume the communication channel model is first-order Markovian as in [30].

With Channel State Information (CSI) known, each user i submits an auction bid $(\xi_{ti}, \hat{v}_{ti})$ to the base station, where ξ_{ti} is the data rate the user i is requesting, and $\hat{v}_{ti}$ is the user i's claimed true value over ξ_{ti}. "True value" is a term in auction theory, and it is denoted as v_{ti} for each user in each scheduling slot. In this work, a user's true value is defined as the instantaneous VR video watching reward as $v_{ti} = r_{tik}$. After receiving the bid from all users, the base station decides the auction winner based on the Vickrey–Clarke–Groves (VCG) auction mechanism, whose following manner makes it ideal for communication resource allocation.

- *Efficiency* - When all users announce their real true values, the overall true value is maximized by efficient communication resource allocation.
- *Individual Rationality* - Each user can expect a non-negative payoff $\hat{v}_{ti} - \sigma_{ti}$ at any scheduling slot t.
- *Truthfulness* - No user can improve its payoff by bidding differently from its true value, which implies that the optimal bid at any scheduling slot is $\hat{v}_{ti} = v_{ti}$.

In VCG, the base station determines the auction winner to maximize the overall claimed true value. We denote by $\boldsymbol{\psi}_t \in \{0,1\}^N$ the auction winner determination and $\boldsymbol{\phi}_t \in \{0,1\}^{S\times N}$ the resource block allocation, where $\psi_{ti} = 1$ means the user i wins resource blocks to support the data rate it requests, and $\phi_{tji} = 1$ means the resource block j is allocated to the user i. The winner determination can be derived as (10).

$$\begin{aligned} &\max_{\boldsymbol{\phi}_t} \sum_{i=1}^{N} \psi_{ti}\cdot\hat{v}_{ti} \\ &s.t.\ \psi_{ti} = \begin{cases} 1, \sum_{j=1}^{S}\phi_{tji} b^p_{tji} \geqslant \xi_{ti} \\ 0, \sum_{j=1}^{S}\phi_{tji} b^p_{tji} < \xi_{ti} \end{cases} \\ &\sum_{i=1}^{N}\phi_{tji} \leqslant 1 \\ &\phi_{tji} \in \{0,1\} \end{aligned} \tag{10}$$

Note that the allocated resource blocks in (10) might be fewer than the total available resource blocks, while the remaining resource blocks are insufficient for any losing user. In order to preserve the efficiency of the VCG mechanism, the remaining resource blocks cannot be allocated to losing users; otherwise, users may try to lose the VCG auction and utilize the freely allocated resources. As for the remaining resource blocks, they can be scheduled to serve other applications. Moreover, since the remaining resources are insufficient for any single losing user's requested data rate, this inefficiency can be neglected as the total communication resources increase.

With the communication resource block allocation and winner determination, the user i's available data rate can be given by (11).

$$b_{ti} = \psi_{ti}\xi_{ti} \tag{11}$$

Moreover, in the VCG auction, the user who obtains requested resource blocks needs to make a payment σ_{ti} to the base station, and the payment is calculated as

$$\sigma_{ti} = \max_{\boldsymbol{\psi}_{-i}} \sum_{i'=1,i'\neq i}^{N} \psi_{ti'}\cdot\hat{v}_{ti'} - \max_{\boldsymbol{\psi}} \sum_{i'=1,i'\neq i}^{N} \psi_{ti'}\cdot\hat{v}_{ti'} \tag{12}$$

where $\boldsymbol{\psi}_{-i}$ is the VCG winner determination when user i does not participate in the auction, and $\boldsymbol{\psi}$ is VCG winner determination when user i participates in the auction. In short, the payment made by user i is the true value loss it causes to other users.

With the VCG auction, the communication resource block allocation can achieve the following properties.

IV. ComPer-FedAvg for Viewport Prediction Model Training

This section introduces the design principle and detail of the prediction model training algorithm, i.e., the ComPer-FedAvg algorithm.

$$\min_{\boldsymbol{\omega}^c_k} \mathcal{L}(\boldsymbol{\omega}^c_k) := \min_{\boldsymbol{\omega}^c_k} \left(\frac{\iota^u}{N}\sum_{i=1}^{N} l_i(\boldsymbol{\omega}^c_k - \alpha^p \nabla l_i(\boldsymbol{\omega}^c_k)) + \frac{\iota^c}{N}\sum_{i=1}^{N} l_{jk}(\boldsymbol{\omega}^c_k - \alpha^p \nabla l_i(\boldsymbol{\omega}^c_k)) \right) \tag{7}$$

$$\nabla\mathcal{L}_i(\boldsymbol{\omega}^c_k) = \left(I - \alpha^p\nabla^2 l_i(\boldsymbol{\omega}^c_k)\right)\left(\iota^u \nabla l_i(\boldsymbol{\omega}^c_k - \alpha^p\nabla l_i(\boldsymbol{\omega}^c_k)) + \iota^c \nabla l_{jk}(\boldsymbol{\omega}^c_k - \alpha^p \nabla l_i(\boldsymbol{\omega}^c_k))\right) \tag{8}$$

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

A. ComPer-FedAvg Learning Objective

When different users watch different VR videos, two features can be leveraged to help predict viewports: the user's personal viewing pattern and the video's common viewing pattern. The personal viewing pattern reflects the user's preferred head movement over a given viewport. The video's common viewing pattern reflects the video's viewing feature shared among all its viewers.

User i's personal viewing pattern can be learned from its historical viewing history $\boldsymbol{\mathcal{D}}_i^u$, and the VR video k's common viewing pattern can be learned from the VR video k's viewing histories $\boldsymbol{\mathcal{D}}_{jk}^c$ stored in the CP j.

When training the personalized prediction model, the central training node balances between users' personal viewing pattern and the VR video's common viewing pattern, which is denoted as in (7). In (7), $\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c)$ as in (2) is the personalized weight at each user. $\frac{\iota^u}{N}\sum_{i=1}^{N} l_i(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c))$ is the personalized model's average loss on every user's data set, which characterizes how the federally learned model caters to each user's personal viewing pattern. Meanwhile, $\frac{\iota^c}{N}\sum_{i=1}^{N} l_{jk}(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c))$ is the personalized model's loss on the VR video k's existing viewing history, which characterizes how the model caters to the VR video k's viewing pattern. Besides, ι^u and ι^c is the weight parameter that balances the model's preference over users' personal and VR video's common viewing pattern.

B. ComPer-FedAvg Algorithm

Inspired by the Federated Meta-Learning [27] and Personalized Federated Learning (Per-FedAvg) [26] solutions, we propose the ComPer-FedAvg algorithm to solve the problem (7).

In (7), $\mathcal{L}(\boldsymbol{\omega}_k^c)$ can be written as the average of *meta-function* $\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_N$, where $\mathcal{L}_i$ is associated with the user i as in (13).

$$\mathcal{L}_i(\boldsymbol{\omega}_k^c) = \iota^u l_i(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c)) + \iota^c l_{jk}(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c)) \tag{13}$$

Then to solve problem (7), the first step is to compute the gradient of meta-functions $\nabla \mathcal{L}_i$, which is given by (8).

The important task in (8) is to compute the gradient $\nabla l_i(\boldsymbol{\omega}_k^c)$ and the Hessian $\nabla^2 l_i(\boldsymbol{\omega}_k^c)$, which is computation costly. Hence, we take a batch of data $\boldsymbol{D}_i^u \in \boldsymbol{\mathcal{D}}_i^u$ with respect to the distribution p_i to obtain an unbiased estimate of the gradient $\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c, \boldsymbol{D}_i^u)$ as given by (14).

$$\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c, \boldsymbol{D}_i^u) := \frac{1}{|\boldsymbol{D}_i^u|} \sum_{(\boldsymbol{V}_{tik}, \boldsymbol{V}_{(t+1)ik}) \in \boldsymbol{D}_i^u} \nabla e_i(\boldsymbol{\omega}_k^c; \boldsymbol{V}_{tik}, \boldsymbol{V}_{(t+1)ik}) \tag{14}$$

Computing the Hessian $\nabla^2 l_i(\boldsymbol{\omega}_k^c)$ requires a much higher computation cost. Therefore, extending the result and proof in [31], the Hessian $\nabla^2 l_i(\boldsymbol{\omega}_k^c) \cdot \nabla l_i(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c))$ and $\nabla^2 l_i(\boldsymbol{\omega}_k^c) \cdot \nabla l_{jk}(\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c))$ in (8) can be approximated as in (15) and (16), where $\boldsymbol{D}_i^u$, $\boldsymbol{D}_i^{u\prime}$, $\boldsymbol{D}_i^{u\prime\prime}$ are different mini-batches from user i's personal viewing history, and $\boldsymbol{D}_{jk}^c \in \boldsymbol{\mathcal{D}}_{jk}^c$ is a mini-batch from the CP j's history on VR video k.

Algorithm 1 The proposed ComPer-FedAvg algorithm.

Input: Initial iterate $\boldsymbol{\omega}_{k(0)}^c$, fraction ζ of active users participating in the training process.

1: **for** Iteration number κ^1: 1 to K^1 **do**
2: The central node chooses a subset of users $\boldsymbol{\mathcal{F}}_{\kappa^1}$ uniformly at random with size ζN.
3: The central node sends $\boldsymbol{\omega}_{k(\kappa^1)}^c$ to all users in $\boldsymbol{\mathcal{F}}_{\kappa^1}$.
4: **for** all $i \in \boldsymbol{\mathcal{F}}_{\kappa^1}$ **do**
5: Set $\boldsymbol{\omega}_{ik(\kappa^1,0)}^u = \boldsymbol{\omega}_k^c$.
6: **for** κ^2: 0 to K^2 **do**
7: Compute the mini-batch gradient descent $\widetilde{\nabla} l_i(\boldsymbol{\omega}_{ik(\kappa^1,\kappa^2-1)}^u, \boldsymbol{D}_{i(\kappa^2)}^u)$ using dataset $\boldsymbol{D}_{i(\kappa^2)}^u$.
8: Set $\widetilde{\boldsymbol{\omega}}_{ik(\kappa^1,\kappa^2)}^u = \boldsymbol{\omega}_{ik(\kappa^1,\kappa^2-1)}^u - \alpha^p \widetilde{\nabla} l_i(\boldsymbol{\omega}_{ik(\kappa^1,\kappa^2-1)}^u, \boldsymbol{D}_{i(\kappa^2)}^u)$.
9: Send $\widetilde{\boldsymbol{\omega}}_{ik(\kappa^1,\kappa^2)}^u$ to the central node, the central node calculates $\widetilde{\nabla} l_{jk}(\widetilde{\boldsymbol{\omega}}_{ik(\kappa^1,\kappa^2)}^u, \boldsymbol{D}_{jk(\kappa^1,\kappa^2)}^c)$, and sends it back to user i.
10: Set $\boldsymbol{\omega}_{ik(\kappa^1,\kappa^2)}^u$ as in (17) with the approximation in (15) and (16).
11: **end for**
12: User i sends $\boldsymbol{\omega}_{ik(\kappa^1,K^2)}^u$ to the central node.
13: **end for**
14: The central node updates its model by averaging over received models $\boldsymbol{\omega}_{jk(\kappa^1)}^c = \frac{1}{N\zeta}\sum_{i\in\boldsymbol{\mathcal{F}}_k} \boldsymbol{\omega}_{ik(\kappa^1,K^2)}^u$.
15: **end for**

With the approximated gradient and Hessian calculated, we present the procedure of the proposed ComPer-FedAvg algorithm in Algorithm 1. The algorithm proceeds in an iterative manner. For the central node, in step 1 to 3, it initiates a central weight $\boldsymbol{\omega}_{k(0)^c}$, selects a random subset of users to participate in the ComPer-FedAvg process, and sends the central weight $\boldsymbol{\omega}_{k(\kappa^1)^c}$ to the participating users. Then for each participating user, they initiate their local weights $\boldsymbol{\omega}_{ik}^u$ as the central weight in step 5, and they carry out K^2 steps of local mini-batch gradient descent in step 6 to 11. More specifically, in step 7, the user i's gradient descent on $\boldsymbol{\omega}_{ik}^u$ is estimated with $\widetilde{\nabla} l_i(\boldsymbol{\omega}_{ik(\kappa^1,\kappa^2-1)}^u, \boldsymbol{D}_{i(\kappa^2)}^u)$. In step 8, $\widetilde{\boldsymbol{\omega}}_{ik}^u$ denotes the locally personalized weight $\boldsymbol{\omega}_{ik}^u$, i.e., the local training version of $\boldsymbol{\omega}_k^c - \alpha^p \nabla l_i(\boldsymbol{\omega}_k^c)$ in (2). In step 9, the user sends $\widetilde{\boldsymbol{\omega}}_{ik}^u$ to the central node. With $\widetilde{\boldsymbol{\omega}}_{ik}^u$ received, the central node computes the estimated gradient $\widetilde{\nabla} l_i(\boldsymbol{\omega}_{ik(\kappa^1,\kappa^2-1)}^u, \boldsymbol{D}_{i(\kappa^2)}^u)$ and returns it to the user. In step 10, the local training weight is updated by the mini-batch gradient descent as in (8) with the approximation described in (15) and (16). After the local iteration terminates, users send their local weights to the central node in step 12, and the central node averages over of the received weights in step 14.

V. VR Video Streaming Problem Formulation

In this section, we formulate the communication resource block allocation problem among the users across the time horizon as a stochastic game. Then we discuss the best-response solution from a game-theoretic perspective.

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

$$\nabla^2 l_i(\boldsymbol{\omega}_k^c)\cdot\nabla l_i(\boldsymbol{\omega}_k^c-\alpha^p\nabla l_i(\boldsymbol{\omega}_k^c))\approx \frac{\widetilde{\nabla} l_i\left(\boldsymbol{\omega}_k^c+\delta\widetilde{\nabla} l_i\big(\boldsymbol{\omega}_k^c-\alpha^p\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c,\boldsymbol{D}_i^u),\boldsymbol{D}_i^{u\prime}\big),\boldsymbol{D}_i^{u\prime\prime}\right)-\widetilde{\nabla} l_i\left(\boldsymbol{\omega}_k^c-\delta\widetilde{\nabla} l_i\big(\boldsymbol{\omega}_k^c-\alpha^p\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c,\boldsymbol{D}_i^u),\boldsymbol{D}_i^{u\prime}\big),\boldsymbol{D}_i^{u\prime\prime}\right)}{2\delta} \tag{15}$$

$$\nabla^2 l_i(\boldsymbol{\omega}_k^c)\cdot\nabla l_{jk}(\boldsymbol{\omega}_k^c-\alpha^p\nabla l_i(\boldsymbol{\omega}_k^c))\approx \frac{\widetilde{\nabla} l_i\left(\boldsymbol{\omega}_k^c+\delta\widetilde{\nabla} l_{jk}\big(\boldsymbol{\omega}_k^c-\alpha^p\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c,\boldsymbol{D}_i^u),\boldsymbol{D}_{jk}^c\big),\boldsymbol{D}_i^{u\prime\prime}\right)-\widetilde{\nabla} l_i\left(\boldsymbol{\omega}_k^c-\delta\widetilde{\nabla} l_{jk}\big(\boldsymbol{\omega}_k^c-\alpha^p\widetilde{\nabla} l_i(\boldsymbol{\omega}_k^c,\boldsymbol{D}_i^u),\boldsymbol{D}_{jk}^c\big),\boldsymbol{D}_i^{u\prime\prime}\right)}{2\delta} \tag{16}$$

$$\boldsymbol{\omega}^u_{ik(\kappa^1,\kappa^2)}=\boldsymbol{\omega}^u_{ik(\kappa^1,\kappa^2-1)}-\alpha^t\left(I-\alpha^p\widetilde{\nabla}^2 l_i(\boldsymbol{\omega}^u_{ik(\kappa^1,\kappa^2-1)},\boldsymbol{D}^{u\prime}_{i(\kappa^2)})\right)\left(\iota^u\widetilde{\nabla} l_i(\widetilde{\boldsymbol{\omega}}^u_{ik(\kappa^1,\kappa^2)},\boldsymbol{D}^{u\prime\prime}_{i(\kappa^2)})+\iota^c\widetilde{\nabla} l_{jk}(\widetilde{\boldsymbol{\omega}}^u_{ik(\kappa^1,\kappa^2)},\boldsymbol{D}^c_{jk(\kappa^1,\kappa^2)})\right) \tag{17}$$

A. Stochastic Game Formulation

Due to limited resource blocks and the stochastic nature in the networking environment and user viewport prediction, we formulate the communication resource allocation problem into a stochastic game among VR video users. VR video users are competitive players competing for communication resource blocks.

Denote the overall state space at the t-th scheduling slot as $\boldsymbol{\mathcal{S}}_t=[\boldsymbol{\mathcal{S}}_{t1},\boldsymbol{\mathcal{S}}_{t2},\cdots,\boldsymbol{\mathcal{S}}_{tN}]$, where $\boldsymbol{\mathcal{S}}_{ti}$ is the local state of the user i at the t-th scheduling slot. More specifically, the state space of the user i consists of the information on the buffered tiles for the current viewport, and the data rate it can achieve on each resource block, as is defined in (18).

$$\boldsymbol{\mathcal{S}}_{ti}=[\boldsymbol{B}^H_{tik},\boldsymbol{B}^L_{tik},b^p_{t1i},\cdots,b^p_{tSi}] \tag{18}$$

Meanwhile, the joint control policies of users are denoted as $\boldsymbol{\pi}=[\pi_1,\pi_2,\cdots,\pi_N]$, where π_i, is the user i's control strategy. For the individual user, as is shown in (19), its strategy is a probability distribution over the actions $\mathcal{A}_{ti}$ it can take, which is dependent on the state it observes. In (19), $\mathbb{P}(\cdot)$ is the probability of an event. As for each action $\mathcal{A}_{ti}$, it is to decide the data rate the user should request and how much it should claim as its true value for the requested data rate, as defined in (20).

$$\mathbb{P}(\mathcal{A}_{ti})=\pi_i(\boldsymbol{\mathcal{S}}_{ti}) \tag{19}$$

$$\mathcal{A}_{ti}=(\xi_{ti},\hat{v}_{ti}) \tag{20}$$

At the beginning of each scheduling slot, each player observes the overall state $\boldsymbol{\mathcal{S}}_t$, takes its action $\mathcal{A}_{ti}$ according to $\pi_i(\boldsymbol{\mathcal{S}}_t)$, and receives an immediate utility $\mathcal{U}_{ti}$. According to the VCG mechanism, the user needs to pay for the resources it achieves as in (12). Therefore, the immediate utility is composed of the reward the user achieves from viewing the VR video (6) and the payment it makes for the requested data rate, which is shown in (21).

$$\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t,\mathcal{A}_{ti},\mathcal{A}_{t(-i)})=r_{tik}-\sigma_{ti} \tag{21}$$

With the assumption on the random communication channel quality and the prediction accuracy, the stochastic game's state transition can be represented as (22).

$$\mathbb{P}(\boldsymbol{\mathcal{S}}_{t+1}|\boldsymbol{\mathcal{S}}_t,\boldsymbol{\pi})=\prod_{i=1}^{N}\mathbb{P}\left([\boldsymbol{B}^H_{(t+1)ik},\boldsymbol{B}^L_{(t+1)ik}]\middle|[\boldsymbol{B}^H_{tik},\boldsymbol{B}^L_{(t+1)ik}],\boldsymbol{\pi}\right)\cdot\prod_{j=1}^{S}\mathbb{P}\left(b^p_{(t+1)ji}\middle|b^p_{tji}\right) \tag{22}$$

Intuitively, the user's goal is to maximize their overall utility by taking actions $\mathcal{A}_{ti},t=1,2,\cdots$ as in (23). However, to achieve the optimal overall utility, users need to know the exact state information during the whole watching period, which is not available for them when deciding the communication resource bidding. Therefore, when making decisions at each scheduling slot, the users' objective is to maximize their expected long-term utility $\mathcal{V}_i(\boldsymbol{\mathcal{S}}_t,\boldsymbol{\pi})$ as shown in (24). In (24), $\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t,\pi_i,\pi_{-i})$ is the utility the user i can achieve when it plays the strategy π_i and other users play their strategy π_{-i}. Meanwhile, γ is the discounting factor reflecting the user's preference over immediate and future utility, which indicates that the further the utility is, the less important it is for the current decision.

$$\max_{\mathcal{A}_{ti}}\sum_{t=1,2,\cdots}\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t,\mathcal{A}_{ti},\mathcal{A}_{t(-i)}) \tag{23}$$

$$\mathcal{V}_i(\boldsymbol{\mathcal{S}}_t,\boldsymbol{\pi})=\mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=1}^{\infty}\gamma^{(t-1)}\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t,\pi_i,\pi_{-i})\right] \tag{24}$$

The long-term utility can also be termed as the state value function, and the aim of each user in the stochastic game is to find a strategy $\pi_i^\star$ that maximizes their state value function in each step, which can be formally formulated as

$$\pi_i^\star=\arg\max_{\pi_i}\mathcal{V}_i(\boldsymbol{\mathcal{S}},\pi_i,\pi_{-i}),\forall\boldsymbol{\mathcal{S}},\pi_{-i} \tag{25}$$

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

A Nash equilibrium (NE) can describe users' rational behavior in the formulated stochastic game.

Definition 1: In the formulated stochastic game, an NE is a condition where each user has a best response strategy $\pi_i^\star$ when all other players play their best response $\pi_{-i}^\star$ strategy.

Note that, for the N-player stochastic game with expected infinite-horizon discounted payoffs, there always exists an NE in stationary control strategies [32]. Therefore, as can be easily observed from (24), there is an NE for the formulated stochastic game.

B. Best Response of the Stochastic Game

Supposing in the stochastic game, the user has a perfect information on the global network state transition probability, and all users play the NE strategy $\boldsymbol{\pi}^\star$, then the optimal state value function $\mathcal{V}_i^\star(\boldsymbol{\mathcal{S}}_t)$ can be derived by (26).

$$\begin{aligned}\mathcal{V}_i^\star(\boldsymbol{\mathcal{S}}_t) = \max_{\pi_i} \Big\{ &\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \pi_i, \pi_{-i}^\star) + \\ &\gamma \sum_{\boldsymbol{\mathcal{S}}_{t+1}} \mathbb{P}(\boldsymbol{\mathcal{S}}_{t+1} | \boldsymbol{\mathcal{S}}_t, \pi_i, \pi_{-i}^\star) \cdot \mathcal{V}_i^\star(\boldsymbol{\mathcal{S}}_{t+1}) \Big\}\end{aligned} \tag{26}$$

Meanwhile, the NE strategy for each user can be expressed as $\pi_i^\star = \arg\max_{\pi_i} \mathcal{V}_i(\boldsymbol{\mathcal{S}})$. However, since the priori state transition probability is unknown to users, it is challenging to directly derive the optimal strategy.

C. POMDP Reformulation

Besides the lack of system dynamics, in the competitive communication resource auction, users have no incentive to share their private information with others. Therefore, each user can only observe their local state and take actions accordingly. However, as shown in (21), the utility is decided by both the overall state and action. The inconsistency between the global state and local observation makes the problem a POMDP problem, which can be denoted by $(\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, \mathcal{P}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{O}}, \Omega)$. $\boldsymbol{\mathcal{S}}$, $\boldsymbol{\mathcal{A}}$, $\mathcal{P}$, and $\boldsymbol{\mathcal{U}}$ are the global state, actions, state transition probability, and utility in the stochastic game. $\boldsymbol{\mathcal{O}}$ is the observation space, where $\boldsymbol{\mathcal{O}}_{ti}$ is the user i's observation at the t-th scheduling slot. The observation is generated from the global state according to the rule Ω, i.e., $\boldsymbol{\mathcal{O}}_{ti} = \Omega(\boldsymbol{\mathcal{S}}_t)$. In this work, the observation is the same as the user's local state, which is expressed as $\boldsymbol{\mathcal{O}}_{ti} = \mathcal{S}_{ti}$.

In the reformulated POMDP, users' objective is to find a policy $\hat{\pi}_i^\star$ that maximizes their observation based value function in each step, which is shown in (27).

$$\begin{aligned}&\max_{\hat{\pi}_i} \mathcal{V}_i(\boldsymbol{\mathcal{O}}_t, \hat{\pi}_i) := \\ &\max_{\hat{\pi}_i} \mathbb{E}_{\hat{\boldsymbol{\pi}}} \left[\sum_{\tau=t}^{\infty} \gamma^{(\tau-t)} \mathcal{U}_{ti} \Big(\boldsymbol{\mathcal{S}}_\tau, \hat{\pi}_i(\boldsymbol{\mathcal{O}}_{\tau i}), \hat{\pi}_{-i}(\boldsymbol{\mathcal{O}}_{\tau(-i)}) \Big) \right]\end{aligned} \tag{27}$$

VI. VR Video Streaming Solution

In this section, we solve the stochastic game from two perspectives, i.e., how much data rate to request for each user and how much to claim as their true value. For the true value claiming problem, we analyze the optimal claiming strategy for each user. Meanwhile, for the resource block requesting problem, we elaborate on how users can take advantage of DRL to achieve optimal VR video watching in the stochastic game.

A. Bidding Strategy Analysis

In the communication resource auction process, the user submits two bids to the base station, namely, the requested data rate and the true value it achieves on the requested true value. Considering the payment to be made, the user cannot request too much data rate; otherwise, the potential payment could overwhelm the reward it achieves. Also, when reporting the true value to the base station, it needs to decide whether to deviate from its real true value.

Lemma 1: For non-decreasing true value over requested data rate, the maximum overall true value loss in the VCG auction caused by user i is its claimed true value $\hat{v}_{ti}$.

Proof: Considering the winner determination mechanism in the VCG auction, the central auction node decides whether the user wins the auction by maximizing the overall true value. Therefore, if user i wins the auction, (28) must be met with $\psi_{ti} = 1$, which means the central auction node can achieve a higher overall true value by letting the user i win. Then we can derive that the maximum true value loss caused by user i is $\hat{v}_{ti}$ as in (29).

$$\max_{\boldsymbol{\psi}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} + \psi_{ti} \hat{v}_{ti} \geqslant \max_{\boldsymbol{\psi}_{-i}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} \tag{28}$$

$$\sigma_{ti} = \max_{\boldsymbol{\psi}_{-i}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} - \max_{\boldsymbol{\psi}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} \leqslant \hat{v}_{ti} \tag{29}$$

Theorem 1: In the proposed system, after the user decides the data rate to request, its dominant strategy is to claim its true value $\hat{v}_{ti}$ as the real true value v_{ti} on the requested data rate.

Proof: Suppose the user wins the auction with the claimed true value $\hat{v}_{ti}$ and makes a payment σ_{ti}. As is proved in (29), $\sigma_{ti} \leqslant \hat{v}_{ti}$. If the payment σ_{ti} (or the overall true value loss caused by user i) is lower than user i's real true value v_{ti}, then user i can also win the auction with the claimed value $\hat{v}_{ti} = v_{ti}$ as in (30), so there is no need to claim a higher true value in this case. On the contrary, if the payment is higher than v_{ti}, then the utility of the user $v_{ti} - \sigma_{ti}$ will be negative, which means it is not a worthy option for the user to win the auction. Therefore, users have no desire to claim higher than their real true value. Moreover, claiming a lower true value cannot bring users any extra utility, either. The reason is that as long as the claimed true value $\hat{v}_{ti}$ is higher than the overall true value loss σ_{ti}, the user cannot reduce the payment

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

$$Q_i^{\star}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) = \mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) + \gamma \sum_{\boldsymbol{\mathcal{S}}_{t+1}} \mathbb{P}(\boldsymbol{\mathcal{S}}_{t+1}|\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) \cdot \max_{\mathcal{A}_{(t+1)i}} Q_i^{\star}(\boldsymbol{\mathcal{S}}_{t+1}, \mathcal{A}_{(t+1)i}, \mathcal{A}_{(t+1)(-i)}) \quad (32)$$

by reducing the claimed true value. When $\hat{v}_{ti} \leqslant \sigma_{ti}$, the user cannot win the auction with the claimed $\hat{v}_{ti}$. As a result, users have no will to claim a lower true value, either.

$$\begin{aligned} &\sigma_{ti} = \max_{\boldsymbol{\psi}_{-i}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} - \max_{\boldsymbol{\psi}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} \leqslant v_{ti} \\ &\rightarrow \max_{\boldsymbol{\psi}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} v_{ti'} + \psi_{ti} \hat{v}_{ti} \geqslant \max_{\boldsymbol{\psi}_{-i}} \sum_{i'=1, i' \neq i}^{N} \psi_{ti'} \hat{v}_{ti'} \end{aligned} \quad (30)$$

With the conclusion above, the joint resource block requesting and true value claiming policy can be decomposed into two disjoint phases, i.e., the optimal data rate requesting and the real true value reporting. Since the user can easily evaluate the true value it could achieve with given data rate based on (4) (5) (6) (11), we focus on the optimal data rate requesting in the following.

B. Q-Learning for the Stochastic Game

Suppose users have perfect information about the system state, then they request the optimal number of resource blocks to maximize their long-term utility at each scheduling slot as in (24). Considering the lack of priori information on the system dynamics, Q-learning [33] based approach can be utilized to learn the optimal policy for users.

Rather than learning the state value function $\mathcal{V}_i(\boldsymbol{\mathcal{S}}_t, \boldsymbol{\pi})$, Q-learning learns the action value function $Q(\cdot)$ shown in (31).

$$\begin{aligned} &Q_i(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) = \mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) + \\ &\gamma \sum_{\boldsymbol{\mathcal{S}}_{t+1}} \mathbb{P}(\boldsymbol{\mathcal{S}}_{t+1}|\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) Q_i(\boldsymbol{\mathcal{S}}_{t+1}, \mathcal{A}_{(t+1)i}, \mathcal{A}_{(t+1)(-i)}) \end{aligned} \quad (31)$$

According to the Bellman optimality equation [34], the optimal action value function can be denoted as in (32). To achieve the optimal long-term reward in each state, the user only needs to find the action $\mathcal{A}_{ti}^{\star}$ that maximizes the action value function $Q_i^{\star}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)})$.

Q-learning learns the optimal state value function in an iterative off-policy manner [33]. More specifically, the learning agent continuously interacts with the environment, observes an environment state $\boldsymbol{\mathcal{S}}_t$, takes an action $\mathcal{A}_{ti}$, receives an instantaneous utility $\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)})$, and observes the environment transits to a new state $\boldsymbol{\mathcal{S}}_{t+1}$. The above interaction is denoted as an experience memory entity $(\boldsymbol{\mathcal{S}}_t, \boldsymbol{\xi}_t, \mathcal{U}_{ti}, \boldsymbol{\mathcal{S}}_{t+1}) \in \boldsymbol{\mathcal{M}}_i$, where $\boldsymbol{\mathcal{M}}_i$ is the experience memory maintained by the learning agent or user i. The learning agent uses the experience above to update its estimation of action value functions, and the update rule is shown in (33). In (33), η is the step size of the learning process. Q-learning converges to the optimal if: a) the local network state transition probability is stationary; and b) all state-action pairs are visited infinitely often [35]. Condition b) can be satisfied when the probability of choosing any action in any state is non-zero, and this is usually achieved by the ϵ-greedy policy. More specifically, when the learning agent interacts with the environment, it chooses the action $\mathcal{A}_{ti}$ that maximizes the currently estimated $Q_i(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)})$ with probability $0 < \epsilon < 1$, and chooses an arbitrary action with possibility $1 - \epsilon$. Meanwhile, in Q-learning, the off-policy learning approach adopts the greedy policy to estimate the action value function of the next state, which is reflected in $\max_{\mathcal{A}_{(t+1)i}} Q_i(\boldsymbol{\mathcal{S}}_{t+1}, \mathcal{A}_{(t+1)i}, \mathcal{A}_{(t+1)(-i)})$.

With the optimal true value claim known, the users only need to decide how much data rate to request. Therefore, the actions $\mathcal{A}_{ti}$ and $\mathcal{A}_{t(-i)}$ in (31), (32), and (33) can be replaced by ξ_{ti} and $\xi_{t(-i)}$.

C. D3RQN for POMDP-based Resource Block Requesting

For the formulated stochastic game, Q-learning can be adopted to solve it. However, conventional Q-learning requires the learning agent to maintain a Q-table to store the state-action pairs and their action value functions, which is prone to the curse of dimension in terms of state and action. Moreover, conventional Q-Learning does not cater well to POMDP problems. The recent progress in deep reinforcement learning cancels the Q-table and replaces it with a deep neural network, and the deep neural network can be trained to simulate the state-action value functions, which has been proved to be efficient by many excellent works [10]. Besides, Deuling Double Deep Recurrent Q-Network (D3RQN) algorithm [36] has been proved to be efficient to solve POMDP problems thanks to the recurrent neural networks structure [37]. Therefore, in this work, we adopt the D3RQN algorithm [36] to learn the optimal strategy for each user.

Compared to the stochastic game, in POMDP, the action value function is evaluated based on users' observation as

$$\begin{aligned} &Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}) = \mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \xi_{ti}, \xi_{t(-i)}) + \\ &\gamma \sum_{\boldsymbol{\mathcal{S}}_{t+1}} \mathbb{P}(\boldsymbol{\mathcal{S}}_{t+1}|\boldsymbol{\mathcal{S}}_t, \xi_{ti}, \xi_{t(-i)}) Q_i(\boldsymbol{\mathcal{O}}_{(t+1)i}, \xi_{(t+1)i}) \end{aligned} \quad (34)$$

where the new observation $\boldsymbol{\mathcal{O}}_{(t+1)i}$ is generated from the new state $\boldsymbol{\mathcal{S}}_{(t+1)i}$ with the rule Ω. Since users or the learning agent can only observe the local state, the experience memory cannot store the global state information. Therefore, with D3RQN, each learning agent i maintains an experience memory $\boldsymbol{\mathcal{M}}_i$ with the tuples of $(\boldsymbol{\mathcal{O}}_{ti}, \boldsymbol{\xi}_t, \mathcal{U}_{ti}, \boldsymbol{\mathcal{O}}_{t+1})$.

In D3RQN, two deep neural networks of the same structure are maintained by each learning agent i, namely, the primary network with parameter θ_i and the target network with parameter θ_i^-. The two networks are updated asynchronously, which greatly improves the stability of the training process [10]. The

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

$$Q_i^{new}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) \leftarrow Q_i(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) + \eta \cdot \left(\mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) + \gamma \max_{\mathcal{A}_{(t+1)i}} Q_i(\boldsymbol{\mathcal{S}}_{t+1}, \mathcal{A}_{(t+1)i}, \mathcal{A}_{(t+1)(-i)}) - Q_i(\boldsymbol{\mathcal{S}}_t, \mathcal{A}_{ti}, \mathcal{A}_{t(-i)}) \right) \tag{33}$$

Algorithm 2 D3RQN algorithm for VR video streaming.

1: **for** Each VR video user i **do**
2: Randomly initialize the primary network with θ_i and target network with θ_i^-.
3: **end for**
4: **for** Each training episode **do**
5: Initialize the environment $\boldsymbol{\mathcal{S}}$.
6: **for** Each training step t **do**
7: **for** Each user i **do**
8: Observes the system and retrieve $\boldsymbol{\mathcal{O}}_{ti} = \Omega(\boldsymbol{\mathcal{S}}_t)$.
9: Requests a number of communication resource blocks ξ_{ti} with the ϵ-greedy strategy based on its primary network θ_i.
10: Submits the requested block number ξ_{ti} and true value $\hat{v}_{ti}$ to the base station.
11: **end for**
12: The base station allocates the communication resources and charge for the resources based on the VCG auction mechanism (10) and (12).
13: The system state transits to a new state $\boldsymbol{\mathcal{S}}_{t+1}$.
14: **for** Each user i **do**
15: Receives an instantaneous utility as (21).
16: Observe the new system state and retrieve $\boldsymbol{\mathcal{O}}_{(t+1)i} = \Omega(\boldsymbol{\mathcal{S}}_{t+1})$.
17: Store the transition $(\boldsymbol{\mathcal{O}}_{ti}, \boldsymbol{\xi}_t, \mathcal{U}_{ti}, \boldsymbol{\mathcal{O}}_{t+1})$ to its local memory $\boldsymbol{\mathcal{M}}_i$.
18: **end for**
19: **for** Each user i **do**
20: Samples a mini-batch data from $\boldsymbol{\mathcal{M}}_i$.
21: Calculate the loss $\mathcal{L}_i^{D3RQN}(\theta_i)$ as in (35).
22: Calculate the mini-batch gradient of the loss $\nabla\mathcal{L}^{D3RQN}(\theta_i)$ as (37).
23: Update the primary network with mini-batch gradient descent $\theta_i \leftarrow \theta_i - \alpha^D \nabla\mathcal{L}^{D3RQN}(\theta_i)$.
24: **if** Update=True **then**
25: Update the target network θ_i^- with the primary network θ_i.
26: **end if**
27: **end for**
28: **end for**
29: **end for**

deep neural network input is the environment state, and the outputs are action value functions of corresponding actions. In D3RQN, each learning agent's primary network approximates their action value function denoted by $Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}; \theta_i)$. To update the primary network towards a more precise approximation, the following sequence of loss is optimized.

$$\mathcal{L}_i^{D3RQN}(\theta_i) = \mathbb{E}_{(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}, \mathcal{U}_{ti}, \boldsymbol{\mathcal{O}}_{(t+1)i}) \sim \boldsymbol{\mathcal{M}}_i} \left[y_i - Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}; \theta_i) \right]^2 \tag{35}$$

with

$$y_i = \mathcal{U}_{ti}(\boldsymbol{\mathcal{S}}_t, \xi_{ti}, \xi_{t(-i)}) + \gamma Q_i \left(\boldsymbol{\mathcal{O}}_{(t+1)i}, \arg\max_{\xi_{(t+1)i}} Q_i(\boldsymbol{\mathcal{O}}_{(t+1)i}, \xi_{(t+1)i}; \theta_i); \theta_i^- \right) \tag{36}$$

In (35) and (36), y_i is generated by the target network [38]. Also, the expectation is calculated based on the experience memory $(\boldsymbol{\mathcal{O}}_{ti}, \boldsymbol{\xi}_t, \mathcal{U}_{ti}, \boldsymbol{\mathcal{O}}_{t+1})$ experienced by the learning agent. Based on the experience memory, experience replay can be utilized to train the neural networks. More specifically, with mini-batches from the experience memory $\boldsymbol{\mathcal{M}}_i$, the primary network is updated with the mini-batch gradient descent over the loss.

$$\nabla\mathcal{L}^{D3RQN}(\theta_i) = \mathbb{E}_{(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}, \mathcal{U}_{ti}, \boldsymbol{\mathcal{O}}_{(t+1)i}) \sim \boldsymbol{\mathcal{M}}_i} \left[\left(y_i - Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}; \theta_i) \right) \cdot \nabla Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti}; \theta_i) \right] \tag{37}$$

As for the structure of neural networks, there are two important features in D3RQN, which is important in solving the communication resource requesting problem. The first feature is the decomposition of advantage function and state value function in the neural network architecture, which helps the user to learn faster about the advantage of requesting different number of resource blocks.

The last but not least feature is the adoption of recurrent networks. For POMDP problems, users learn their optimal policy without full information of the system state, which could make their estimation of the observation-based action value function $Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti})$ deviate from the real action value function $Q_i(\boldsymbol{\mathcal{S}}_{ti}, \xi_{ti})$. The recurrent neural network structure [37] helps users to make a better estimation, and it narrows the gap between the observation and real action value functions. Hence, the users can achieve a better performance.

The procedure of the D3RQN algorithm is shown in Algorithm 2. In step 1-3, the neural networks of each user's D3RQN are initialized. From step 4 of the algorithm, users interact with the environment and update their parameters θ_i. More specifically, the training is conducted in an episode manner. At the beginning of each episode, the environment is reset. In step 7-10, each user observes the system state, retrieves an observation $\boldsymbol{\mathcal{O}}_{ti}$ and requests a data rate accordingly. In step 12, the base station collects bids from all users and allocates the resources to them. Meanwhile, the corresponding payment is charged. With given resources, the

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

system transits to a new state, which consists of the transition of VR video buffer and communication channel quality. In step 14-17, each user stores the experience memory tuple to their learning memory $\mathcal{M}_i$. Then in step 20-23, each user updates its primary network of the D3RQN algorithm with mini-batch gradient descent. Note that, α^D in step 23 is the learning rate of the D3RQN algorithm. In step 24-26, the target network is updated every several training steps, indicated by the manually set "Updated" flag.

VII. Evaluation and Discussion

In this section, experiments and simulations are conducted to validate the performance of the proposed ComPer-FedAvg algorithm and the proposed D3RQN-based VR video streaming scheme.

A. ComPer-FedAvg Algorithm Performance

We conduct the viewport prediction model training based on the data set collected in [39], which contains 50 users' viewing traces on 10 pieces of VR videos. When conducting the ComPer-FedAvg training process, 6 arbitrary VR video viewers are chosen to participate in the training process for a randomly given VR video. The selected users drop their local viewing history on the given VR video. Besides, a central node is made up by stacking the remaining users' viewing history on the given VR video. In the training process, the central and personalized weights ι^c and ι^u are both set as 0.5.

Four baseline schemes are implemented to validate the improvement of the ComPer-FedAvg algorithm, which is listed as follows.

- *Conventional Per-FedAvg [26]:* The central node leverages the viewing history of participating users to train a prediction model that minimizes the overall loss of users' locally personalized model.
- *Conventional FedAvg [9]:* The central node leverages the viewing history of participating users to train a prediction model that minimizes all participating users' overall loss.
- *Local:* Users leverage their local viewing history to train their own prediction model.
- *Central:* The central node trains a prediction model based on the remaining users' viewing history on the given VR video.

As for the prediction model structure, we adopt a deep neural network with four Convolutionary LSTM (C-LSTM) layers and four Batch Normalization (BN) layers alternatively cascaded as hidden layers. Note that, although this structure takes advantage of the prevalent LSTM and CNN, the proposed ComPer-FedAvg does not require a specific structure, and the improvement can also be generalized to other structures. The detailed parameter setting is listed in table II.

The average prediction accuracy of different schemes is presented in Fig. 4. As is shown in the figure, the proposed scheme not only first converges, but also achieves the highest prediction accuracy among all the presented schemes. The baseline Central scheme achieves the second-highest accuracy, which indicates that the video's common viewing pattern is a more critical factor for the predicted VR video. The baseline Per-FedAvg and the baseline Local scheme achieves similar performance, and the baseline FedAvg algorithm achieves the lowest accuracy. The result in Fig. 4 validates that by leveraging the information stored in both users' personal viewing history and the CP's common viewing history, it is possible to integrate the personal and common viewing pattern to make a better prediction.

TABLE II: ComPer-FedAvg parameters.

Hidden layers	8	Layer composition	C-LSTM $\times$ 4 BN $\times$ 4
C-LSTM kernal number	8	C-LSTM kernal size	3×3
Learning rate	0.05	Batch size	360
Client	50	Video	10
Participating User	6	Activation function	Relu
Prediction window	5	Local gradient descent	3
Global update K^1	100	Local update K^2	10

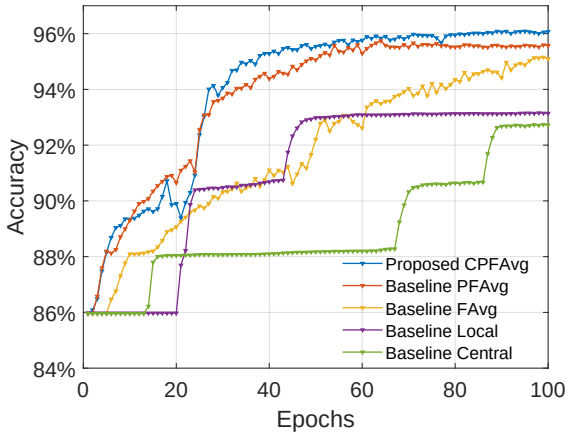


Fig. 4: Average accuracy of the trained prediction model.

Besides the average prediction accuracy, we also present user-wise loss and prediction accuracy in Fig. 5 and Fig. 6. For better visibility, we randomly choose three users to shown the user-wise performance. A promising result indicated by Fig. 5 and Fig. 6 is that the proposed scheme not only achieves better overall performance, it also outperforms baseline solutions in each user. Another observation is that the performance among different users differs slightly. The result indicates that although the trained model has been locally personalized, it cannot perfectly adapt to different user's viewing patterns. The potential reason includes the noise in the training data set and the unpredictable sudden viewport shift of different users.

B. VR Video Streaming Performance

In this subsection, we conduct simulations to validate the performance of the proposed D3RQN-based VR video streaming scheme.

In the simulation, a VR video viewport consists of 20 tiles. Each tile has a high and low-quality version, and their bitrate is set as 500kbps and 100kbps, respectively. The overall bitrate requirement of a given viewport sums up to 2Mbps to 10Mbps [40]. The base station is equipped with 20 resource

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

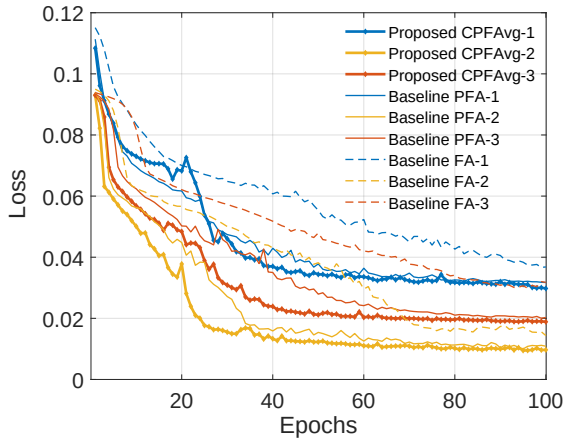


Fig. 5: User-wise loss of the training algorithm.

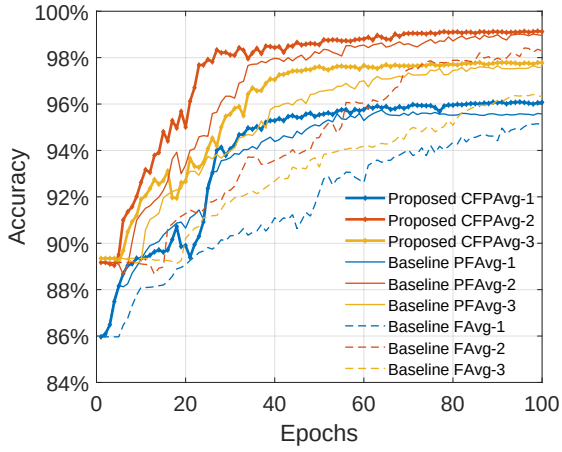


Fig. 6: User-wise accuracy of the trained prediction model.

blocks to allocate, and the data rate each block can provide ranges from 0.2Mbps to 3Mbps. For each user, the data rate transits following a Markov manner. Meanwhile, the transition of different resource block's data rate is independent of each other. In the local network, 3 users compete for the resource blocks to watch their own VR video and prefetch predicted tiles. More specifically, each user observes their local state consisting of local buffer state and data rate resource blocks can provide. With the observation and users' policy, they submit their requested data rate and corresponding true value to the auction node. Then the auction node allocates resource blocks to maximize the overall true value. Note that, the data rate users request are aggregated into discrete values, which ranges from 0.2Mbps to 20Mbps with an interval of 0.2Mbps. Another factor to consider is users' prediction accuracy. For simplicity and better illustration, different users' prediction accuracy is set to follow Gaussian distribution with the average equals 80% and standard deviation equals 5%. The reason to set viewport prediction accuracy to a Gaussian distribution is to quantify its influence on the VR streaming performance in the simulation. As for the implementation of the D3RQN algorithm, the detailed parameters are listed in table III. Note that, in table III, CL is short for Convolutional

TABLE III: D3RQN parameters.

Hidden layers	5	Layer composition	CL $\times$ 3 FLL $\times$ 1 LL $\times$ 1
Activitation function	Relu	Batch size	64
Memory size	10000	Learning rate	0.0001
Utility discounting factor	0.95	Exploration decreasing factor	0.999997
Episode number	500	Steps in each episode	5000

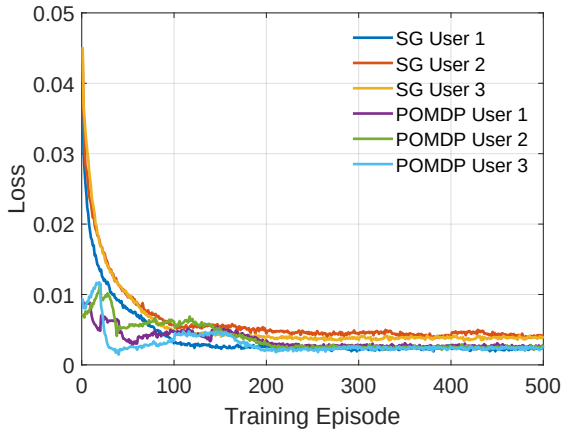


Fig. 7: Convergence of the D3RQN scheme.

Layer, FLL is short for FLatten Layer, and LL is short for LSTM layer.

Apart from the proposed D3RQN-based VR video streaming scheme, three other baseline schemes are implemented to validate the proposed scheme's improvement.

- *ELSN [41]:* Without the local buffer information, the users observe the network state and decide high or low-quality tiles to prefetch. Based on their empirically learned knowledge, they request corresponding resources from the base station.
- *Fair:* The base station arbitrarily allocates the resource blocks to users evenly.
- *Greedy:* Each user observes its local buffer and requests a data rate to achieve its optimal performance.

The convergence of the proposed scheme is presented in Fig. 7 and 8. The baseline scheme is the Stochastic Game (SG), where each user has perfect information of the system state. While in the POMPD case, users can only observe its local state. The abscissa axis in Fig. 7 is the training episode, and in each episode, 500 steps of interaction between the user and the environment are conducted. The vertical axis is the loss defined in (35). As shown in the figure, the D3RQN-based scheme can achieve similar convergence to the stochastic game with perfect global information. All users' loss reduces to a similar level. A similar trend can also be observed in 8. The thin continuous lines are the utilities users can achieve in the SG case, and the thick dotted lines are the utilities users can achieve in the POMDP case. The POMDP case's utilities are a little lower than the SG case, but the gap is within an acceptable margin. The results in 7 and 8 illustrates the D3RQN-based approach can make a very close approximation $Q_i(\boldsymbol{\mathcal{O}}_{ti}, \xi_{ti})$ to $Q_i(\boldsymbol{\mathcal{S}}_t, \boldsymbol{\xi}_t)$.

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

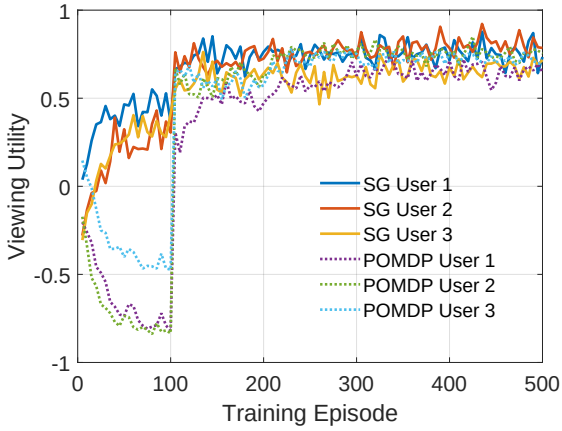


Fig. 8: User-wise streaming utility.

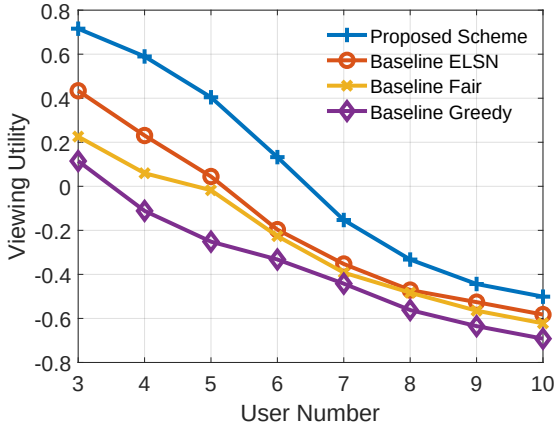


Fig. 10: VR viewing utility v.s. Node number.

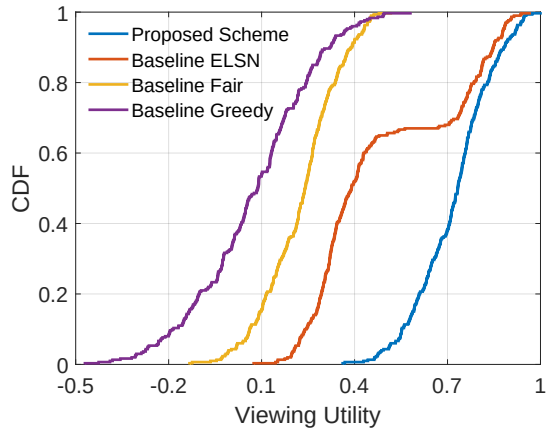


Fig. 9: CDF of VR viewing utility.

In Fig. 9, we present the estimated Cumulative Distribution Function (CDF) of the VR viewing utility for all schemes. The abscissa axis is the episode-wise average VR viewing utility achieved by all users, and the vertical axis is the CDF of the corresponding viewing utility. The VR viewing utility CDF reflects users' VR viewing experience distribution. As is shown in the figure, the proposed scheme achieves the best performance among all presented schemes, and the lowest viewing utility is around 0.3. Meanwhile, the lowest viewing utility of baseline schemes is lower than 0, which means frequent stalls during the VR streaming process. The proposed scheme's better performance in terms of the lowest utility ensures a more consecutive viewing experience for users. As for the overall performance, the baseline ELSN scheme is superior to the baseline fair and greedy scheme. The reason is that the baseline ELSN scheme observes the network state and dynamics and makes tile selection accordingly, which outperforms the simple baseline scheme Fair and Greedy. An interesting observation is that the ELSN scheme achieves uneven performance for different users, which is reflected in the plateau of the ELSN CDF.

The performance of system capacity is shown in 10, where the influence of the different number of users is tested. When the user number increases in the VR streaming system, the performance of all schemes declines. For a given resource supply, the increasing number of users intensifies the competition of resource blocks, and the users cannot retrieve enough data rate to enjoy their VR video. Even though, the proposed scheme achieves the best performance, and it can achieve approximately one more user capacity than the baseline schemes. The baseline ELSN scheme neglects the influence of buffer on VR video viewing, and it achieves the second performance thanks to its efforts in observing the network dynamics and making refined choices on tile quality. The baseline scheme Fair arbitrarily allocates the resource blocks to the user evenly without smart scheduling, making its performance inferior to the more advanced schemes. As for the baseline scheme Greedy, all users request the data rate that maximize their performance. Constrained by the VCG auction mechanism, only a small proportion of users can acquire the requested data rate and it needs to pay a high price. Therefore, the baseline greedy achieves the worst performance among all schemes.

The influence of prediction accuracy is presented in Fig. 11. Intuitively, a higher prediction accuracy helps users to make better use of limited data rate to download tiles that will be watched. And the result in Fig. 11 validates the intuition. As the prediction accuracy increases, the viewing utility achieved by each scheme increases as well. Similar to the case in Fig. 10, the proposed D3RQN scheme achieves the best performance among all schemes. The reason is that the proposed scheme makes resource block requests implicitly considering the prediction accuracy, which is reflected in the buffer state transition. Among all schemes, The proposed scheme is the only one that observes the buffer state and requests resource blocks accordingly.

In Fig. 12, the influence of the overall resource block number is presented. The more resources provided to VR video streaming, the higher possibility that users can acquire the desired resource blocks to achieve their optimal experience. Therefore, all schemes exhibit an increasing trend in the viewing utility. The proposed scheme achieves the highest performance among all schemes. Interestingly, as the

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

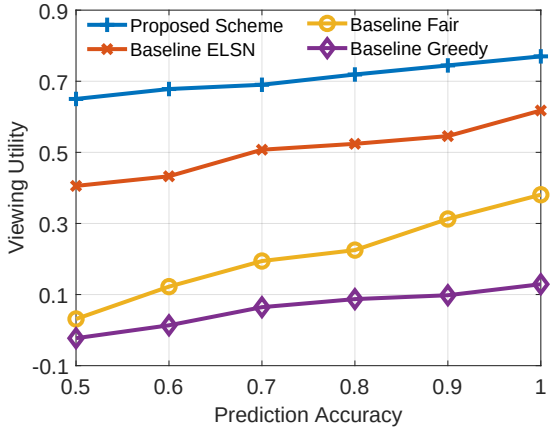


Fig. 11: VR viewing utility v.s. Prediction accuracy.

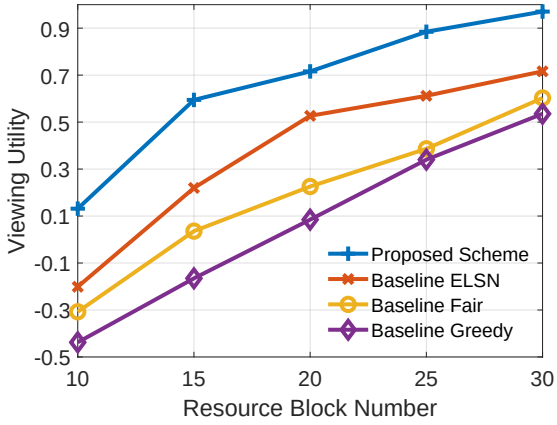


Fig. 12: VR viewing utility v.s. Overall resources.

resource block supply increases to 30 blocks, the baseline scheme Greedy achieves similar performance to the baseline scheme Fair. The reason is as the resources become abundant, it is of higher possibility that users can retrieve the requested resource blocks and thus achieve better performance. Another interesting observation is that the performance increment is more apparent than that brought by improving prediction accuracy as in Fig. 11. The result indicates that although improving the prediction accuracy can improve VR streaming performance, the communication resource provisioning could be a dominant parameter that influences the VR streaming performance.

VIII. Conclusion

This work has proposed an MEC-Enabled VR video streaming system that integrates both VR video viewport prediction and communication resource allocation. The contributions of this work are mainly in two aspects. For the first contribution, considering the importance of viewport prediction in VR video streaming, we have proposed a novel federated learning algorithm to train a viewport prediction model. The proposed ComPer-FedAvg algorithm efficiently leverages the knowledge hidden in users' personal viewing history and a given VR video's common viewing history. Besides, the viewport prediction model is learned in a distributed manner, efficiently protecting users' privacy. The evaluations have proved the improvement of the proposed algorithm. For the second contribution, considering the influence of viewport prediction and prefetch, we have formulated a buffer-aware VR viewing stochastic game to solve the local network's communication resource allocation problem. Considering the privacy concern in the VR viewing process, each VR user distributively decides the data rate to request based on the local network dynamics and current viewing state. With limited information available to each user, we reformulated the stochastic game into a POMDP for each user. The D3RQN algorithm is adopted for each user to achieve their optimal performance. The simulation results validated the improvement of the proposed scheme.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (Grant No.: 2018YFB1800501).

References

[1] M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Toward low-latency and ultra-reliable virtual reality," *IEEE Network*, vol. 32, no. 2, pp. 78–84, 2018.

[2] Y. Zhu, G. Zhai, X. Min, and J. Zhou, "The prediction of saliency map for head and eye movements in 360 degree images," *IEEE Transactions on Multimedia*, vol. 22, no. 9, pp. 2331–2344, 2019.

[3] X. Feng, V. Swaminathan, and S. Wei, "Viewport prediction for live 360-degree mobile video streaming using user-content hybrid motion tracking," *Proc. ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 2, pp. 1–22, 2019.

[4] Q. Yang, J. Zou, K. Tang, C. Li, and H. Xiong, "Single and sequential viewports prediction for 360-degree video streaming," in *Proc. IEEE International Symposium on Circuits and Systems*. IEEE, 2019, pp. 1–5.

[5] C.-L. Fan, S.-C. Yen, C.-Y. Huang, and C.-H. Hsu, "Optimizing fixation prediction using recurrent neural networks for 360° video streaming in head-mounted virtual reality," *IEEE Transactions on Multimedia*, vol. 22, no. 3, pp. 744–759, 2019.

[6] 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 *Proc. ACM International Conference on Multimedia*, 2018, pp. 1190–1198.

[7] 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*, 2020.

[8] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[9] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] M. Chen, O. Semiari, W. Saad, X. Liu, and C. Yin, "Federated echo state learning for minimizing breaks in presence in wireless virtual reality networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 177–191, 2019.

[12] X. Chen, Z. Zhao, C. Wu, M. Bennis, H. Liu, Y. Ji, and H. Zhang, "Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2377–2392, 2019.

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

[13] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *Proc. of the 24th Annual International Conference on Mobile Computing and Networking*, 2018, pp. 99–114.

[14] Y. Ban, L. Xie, Z. Xu, X. Zhang, Z. Guo, and Y. Wang, "Cub360: Exploiting cross-users behaviors for viewport prediction in 360 video adaptive streaming," in *Proc. IEEE International Conference on Multimedia and Expo*. IEEE, 2018, pp. 1–6.

[15] L. Xie, X. Zhang, and Z. Guo, "Cls: A cross-user learning based system for improving qoe in 360-degree video adaptive streaming," in *Proc. ACM International Conference on Multimedia*, 2018, pp. 564–572.

[16] S. Petrangeli, G. Simon, and V. Swaminathan, "Trajectory-based viewport prediction for 360-degree virtual reality videos," in *2018 IEEE International Conference on Artificial Intelligence and Virtual Reality*. IEEE, 2018, pp. 157–160.

[17] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen, "Tiling in interactive panoramic video: Approaches and evaluation," *IEEE Transactions on Multimedia*, vol. 18, no. 9, pp. 1819–1831, 2016.

[18] M. Hosseini, "View-aware tile-based adaptations in 360 virtual reality video streaming," in *Proc. IEEE Virtual Reality*, 2017, pp. 423–424.

[19] N. Q. M. Khiem, G. Ravindra, and W. T. Ooi, "Adaptive encoding of zoomable video streams based on user access pattern," *Signal Processing: Image Communication*, vol. 27, no. 4, pp. 360–377, 2012.

[20] M. Chen, W. Saad, and C. Yin, "Virtual reality over wireless networks: Quality-of-service model and learning-based resource management," *IEEE Transactions on Communications*, vol. 66, no. 11, pp. 5621–5635, 2018.

[21] M. Chen, W. Saad, C. Yin, and M. Debbah, "Data correlation-aware resource management in wireless virtual reality (VR): An echo state transfer learning approach," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4267–4280, 2019.

[22] C. Perfecto, M. S. Elbamby, J. Del Ser, and M. Bennis, "Taming the latency in multi-user VR 360°: A qoe-aware deep learning-aided multicast framework," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2491–2508, 2020.

[23] Y. Sun, Z. Chen, M. Tao, and H. Liu, "Communications, caching, and computing for mobile virtual reality: Modeling and tradeoff," *IEEE Transactions on Communications*, vol. 67, no. 11, pp. 7573–7586, 2019.

[24] T. Dang and M. Peng, "Joint radio communication, caching, and computing design for mobile virtual reality delivery in fog radio access networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 7, pp. 1594–1607, 2019.

[25] S. Gupta, J. Chakareski, and P. Popovski, "Millimeter wave meets edge computing for mobile VR with high-fidelity 8k scalable 360° video," in *Proc. IEEE International Workshop on Multimedia Signal Processing*, 2019, pp. 1–6.

[26] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020.

[27] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," *arXiv preprint arXiv:1802.07876*, 2018.

[28] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited on http://www.cs.toronto.edu/ hinton/coursera/lecture6/lec6.pdf*, vol. 14, no. 8, 2012.

[29] T. C. Nguyen and J.-H. Yun, "Predictive tile selection for 360-degree vr video streaming in bandwidth-limited networks," *IEEE Communications Letters*, vol. 22, no. 9, pp. 1858–1861, 2018.

[30] Y. Chen, H. Tang, J. Wang, and J. Song, "Optimizing age penalty in time-varying networks with markovian and error-prone channel state," *Entropy*, vol. 23, no. 1, p. 91, 2021.

[31] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the convergence theory of gradient-based model-agnostic meta-learning algorithms," in *Proc. International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 1082–1092.

[32] A. M. Fink *et al.*, "Equilibrium in a stochastic n-person game," *Journal of Science of the Hiroshima University, Series AI (mathematics)*, vol. 28, no. 1, pp. 89–93, 1964.

[33] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[34] M. Sniedovich, "A new look at bellman's principle of optimality," *Journal of optimization theory and applications*, vol. 49, no. 1, pp. 161–176, 1986.

[35] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[36] J. Ou, X. Guo, M. Zhu, and W. Lou, "Autonomous quadrotor obstacle avoidance based on dueling double deep recurrent q network with monocular vision," *arXiv preprint arXiv:2002.03510*, 2020.

[37] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable MDPs," in *Proc. AAAI fall symposium series*, 2015.

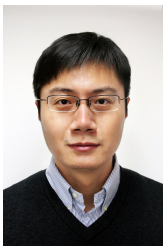
[38] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. International Conference on Machine Learning*. PMLR, 2016, pp. 1995–2003.

[39] W. C. Lo, C. L. Fan, J. Lee, C. Y. Huang, K. T. Chen, and C. H. Hsu, "360° video viewing dataset in head-mounted virtual reality," in *Proc. ACM Multimedia Systems Conference*, 2017, pp. 211–216.

[40] M. Hosseini and V. Swaminathan, "Adaptive 360 VR video streaming: Divide and conquer," in *Proc. IEEE International Symposium on Multimedia*. IEEE, 2016, pp. 107–110.

[41] M. Chen, W. Saad, and C. Yin, "Echo-liquid state deep learning for 360° content transmission and caching in wireless VR networks with cellular-connected UAVs," *IEEE Transactions on Communications*, vol. 67, no. 9, pp. 6386–6400, 2019.

Ran Zhang received his Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2021. He is currently a postdoctoral research fellow in the State Key Laboratory of Networking and Switching Technology, BUPT. His research interests include caching, computing, and communication integration, satellite-terrestrial integrated networks, and artificial intelligence.



Jiang Liu received his B.S. degree in electronics engineering from Beijing Institute of Technology, Beijing, China, in 2005, the M.S. degree in communication and information system from Zhengzhou University, Zhengzhou, China, in 2009, and the Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2012. He is currently an associate professor at Beijing University of Posts and Telecommunications. His current research interests include network architecture, network virtualization, satellite networking, software-defined networking (SDN), information-centric networking (ICN), and network testbed.

Fangqi Liu received her B.S. degree in communication engineering from Jilin University, Jilin, China, in 2020. She is currently pursuing the master's degree in information and communication engineering in the State Key Laboratory of Networking and Switching Technology, BUPT. Her research interests include machine learning, virtual reality, and satellite networking.

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2021.3119144, IEEE Journal on Selected Areas in Communications

Tao Huang received his B.S. degree in communication engineering from Nankai University, Tianjin, China, in 2002, the M.S. and Ph.D. degree in communication and information system from Beijing University of Posts and Telecommunications, Beijing, China, in 2004 and 2007 respectively. He is currently a professor at Beijing University of Posts and Telecommunications. His current research interests include network architecture, network artificial intelligence, routing and forwarding, and network virtualization.

Qinqin Tang received her B.S. degree in information engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2017. She is currently working towards her Ph.D. degree in network engineering in State Key Laboratory of Network and Switching Technology, BUPT. Her research interests include mobile edge networks, traffic offloading, and resource management and allocation.

Shangguang Wang received his Ph.D. degree at Beijing University of Posts and Telecommunications in 2011. He is currently a professor and deputy director at the State Key Laboratory of Networking and Switching Technology, BUPT. He has published more than 100 papers, and played a key role at many international conferences, such as general chair and TPC chair. His research interests include edge computing, service computing, and cloud computing. He is a senior member of the IEEE, and the Editor-in-Chief of the International Journal of Web Science.

F. Richard Yu is a Professor at Carleton University, Canada. His research interests include connected autonomous vehicles, artificial intelligence, and security. He serves on the editorial boards of several journals, including Lead Series Editor for IEEE Trans. Veh. Tech., and IEEE Trans. Green Comm. Net., IEEE Comm. Surveys & Tutorials. He is an IET Fellow, IEEE Fellow, and Distinguished Lecturer of both IEEE Vehicular Technology (VT) Society and Communications Society, the Vice President (Membership) of the IEEE VT Society.

0733-8716 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.