MANSY: Generalizing Neural Adaptive Immersive Video Streaming with Ensemble and Representation Learning

Duo Wu, Panlong Wu, Miao Zhang, *Student Member, IEEE*, and Fangxin Wang, *Member, IEEE*

Abstract—The popularity of immersive videos has prompted extensive research into neural adaptive tile-based streaming to optimize video transmission over networks with limited bandwidth. However, the diversity of users' viewing patterns and Quality of Experience (QoE) preferences has not been fully addressed yet by existing neural adaptive approaches for viewport prediction and bitrate selection. Their performance can significantly deteriorate when users' actual viewing patterns and QoE preferences differ considerably from those observed during the training phase, resulting in poor generalization. In this paper, we propose `MANSY`, a novel streaming system that embraces user diversity to improve generalization. Specifically, to accommodate users' diverse viewing patterns, we design a Transformer-based viewport prediction model with an efficient multi-viewport trajectory input output architecture based on implicit ensemble learning. Besides, we for the first time combine the advanced representation learning and deep reinforcement learning to train the bitrate selection model to maximize diverse QoE objectives, enabling the model to generalize across users with diverse preferences. Extensive experiments demonstrate that `MANSY` outperforms state-of-the-art approaches in viewport prediction accuracy and QoE improvement on both trained and unseen viewing patterns and QoE preferences, achieving better generalization.

Index Terms—tile-based neural adaptive immersive video streaming, generalization, ensemble learning, representation learning

1 Introduction

With the rapid advancement of Virtual Reality (VR) technologies, immersive videos have attracted great attention because of the immersive experience they bring. Recent statistical report [1] projects that the global installed base of VR headsets will exceed 34 million by 2024, marking a remarkable 142% increase since 2020. However, streaming immersive videos is challenging as immersive videos are 4-6 times larger than conventional videos of the same perceived quality because of their panoramic nature [2]. To tackle this issue, tile-based streaming [3] [4] [5] has emerged as an efficient solution to transmit immersive videos to reduce bandwidth consumption and improve Quality of Experience (QoE). In tile-based streaming, video chunks are spatially cropped into non-overlapping tiles, and only tiles inside the user's predicted future viewports are prefetched at high bitrates, thus striking a good balance between bandwidth efficiency and user's QoE.

The implementation of tile-based streaming requires two fundamental building blocks: *viewport prediction* and *bitrate selection*. In recent years, many neural adaptive methods have been proposed to design the two building blocks, which demonstrate superior performance over conventional methods by exploiting the strong non-linear fitting capability of neural networks (NNs). For instance, sequence-to-sequence models such as Gated Recurrent Unit (GRU) [6] and Long Short Term Memeory (LSTM) [7] [8] have showcased higher accuracy for viewport prediction than traditional linear regression [9]. Additionally, deep reinforcement learning (DRL) algorithms [10] [11] [4] have demonstrated more potential for optimizing bitrate selection than heuristic or model-based algorithms as they can determine tile bitrates without any specific presumptions.

Despite promising, existing neural adaptive methods have not fully addressed the challenge of user diversity in *viewing patterns* and *QoE preferences*, leading to poor generalization. On one hand, conventional wisdom trains the viewport prediction model with sets of users' viewport trajectories, but pays little attention to the prediction bias towards the training dataset [7] [10], which may cause significant accuracy loss. Specifically, users' viewing patterns naturally exhibit high diversity [12] [13]. For example, some users may prefer to focus on particular objects, while others may prefer to explore the scene. In this context, when users' actual viewing patterns differ significantly from those observed in the training stage, the model may fail to predict accurately. Such prediction bias limits the model's ability to serve users with diverse viewing patterns, resulting in poor

Manuscript received xxx; revised xxx. This work was supported in part by NSFC with Grant No. 62293482, the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, NSFC with Grant No. 62471423, the Shenzhen Science and Technology Program with Grant No. JCYJ20230807114204010, the Shenzhen Outstanding Talents Training Fund 202002, the Guangdong Research Projects No. 2019CX01X104, the Young Elite Scientists Sponsorship Program of CAST (Grant No. 2022QNRC001), the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001) and the Shenzhen Key Laboratory of Big Data and Artificial Intelligence (Grant No. ZDSYS201707251409055). Fangxin Wang is the corresponding author.

Duo Wu and Panlong Wu are with Shenzhen Future Network of Intelligence Institute (FNii-Shenzhen) and School of Science and Engineering (SSE), The Chinese University of Hong Kong, Shenzhen, China. Duo Wu is also with the Shenzhen International Graduate School, Tsinghua University, China (email: wu-d24@mails.tsinghua.edu.cn;panlongwu@link.cuhk.edu.cn).

Miao Zhang is with the School of Computing Science, Simon Fraser University, BC, Canada (email: mza94@sfu.ca).

Fangxin Wang is with School of Science and Engineering (SSE), Shenzhen Future Network of Intelligence Institute (FNii-Shenzhen), and Guangdong Provincial Key Laboratory of Future Networks of Intelligence, The Chinese University of Hong Kong, Shenzhen, China (email: wangfangxin@cuhk.edu.cn).

generalization.

On the other hand, conventional DRL-based algorithms define reward function as a QoE function with fixed weights to quantify the importance of different QoE metrics (e.g., video quality and rebuffering time) [10] [11]. They then train the DRL model to optimize such function for bitrate selection. Nevertheless, different users often have different QoE preferences [14] [15]. For instance, some users may prioritize high video quality, while others may prioritize smooth playback and tolerate quality distortion. Therefore, it is difficult to characterize users' diverse QoE preferences with a fixed-weight QoE function, as the weights assigned to different QoE metrics can vary significantly among users with different preferences. As a result, the performance of existing DRL algorithms may significantly deteriorate when the optimized QoE function does not align to users' actual preferences [16]. For instance, the DRL model trained to aggressively download low-bitrate tiles to avoid playback interruptions may select inappropriate bitrates for users that prioritize high video bitrates. Consequently, these DRL algorithms fail to generalize across users with diverse QoE preferences.

To tackle the challenge of user diversity, prior studies [10] [13] [17] have attempted to categorize users into dinsinct groups based on their viewing and QoE preferences, and train personalized models for each group. However, this approach necessitates the retraining of a new model whenever a new user group emerges, resulting in prohibitive training cost. As an alternative, researchers in [8] have explored ensemble learning to improve the generalization of viewport prediction models, but their approach involves explicit model duplication, leading to substantial computational overhead. For bitrate selection, recent works [16] [18] have proposed to train the DRL model with multiple QoE functions instead of a single one, but this approach suffers from catastrophic forgetting problem [19] and may still experience performance degradation when serving users with QoE preferences unaligned to those optimized in the training stage.

In this paper, we propose `MANSY`, an ense**M**ble and represent**A**tion lear**N**ing based **SY**stem for tile-based immersive video streaming, which addresses the user diversity challenge to improve generalization. To capture the viewing pattern diversity, we develop a viewport prediction model with an efficient Multi-viewport Trajectory Input Output (MTIO) architecture based on implicit ensemble learning (EL) [20] [21]. The MTIO architecture implicitly trains multiple sub-models with minor computation cost by establishing multiple input-output heads. Each sub-model independently makes prediction, and their prediction results are ensembled to yield well-calibrated predicted viewports that reduce the prediction bias, thus leading to stronger generalization ability. Additionally, we also design the backbone of the prediction model based on Transformer [22], which leverages the attention mechanism to effectively learn long-term dependencies. This enables our model to predict the trends of viewport movements more accurately, further improving the prediction accuracy.

To accommodate users' diverse QoE preferences, we leverage the advanced representation learning (RepL) technique [23] to train the DRL bitrate selection model. Specifically, we encourage the model to mine useful hidden representations from users' QoE preferences, by incorporating mutual information into the reward function for training the model. In this way, we enable our model to capture essential characteristics of users' preferences, such as prioritization of bitrate quality and playback smoothness. This empowers our model to dynamically select bitrates based on users' QoE preferences, even when encountering those unseen during the training phase, thus achieving strong generalization. Additionally, as directly computing mutual information is difficult, we further design an efficient NN model to estimate the mutual information term for reward calculation.

To summarize, this paper makes the following contributions:

- We propose a novel tile-based immersive video streaming system `MANSY` that addresses the challenge of user diversity in both viewing patterns and QoE preferences to significantly improve generalization.
- We design an efficient MTIO-Transformer viewport prediction model based on implicit EL, which effectively reduces the prediction bias to serve users with various viewing patterns with minor computation cost.
- To the best of our knowledge, we are the first to combine RepL and DRL to train the bitrate selection model, enabling the model to maximize users' QoE with diverse preferences and thus achieving better generalization.
- Extensive experiments demonstrate the superiority of `MANSY` in both viewport prediction and bitrate selection. Results indicate that compared to state-of-the-art approaches, `MANSY` improves the mean prediction accuracy by 1.3%-5.2%/3.2%-8.8% and mean QoE by 3.0%-14.1%/3.2%-15.3% on trained/unseen viewing patterns and QoE preferences, respectively.

The rest of this paper is organized as follows. Section 2 presents our observations on the impacts of user diversity, which motivates the design of our work. Section 3 provides the system overview of `MANSY`. Next, we elaborate the detailed design of the MTIO Tranformer viewport prediction model and RepL-enabled bitrate selection model in Section 4 and 5, respectively. We then conduct extensive experiments to evaluate the performance of `MANSY` in Section 6. The related work is provided in Section 7. We also discuss some potential methodologies to further improve `MANSY` in Section 8. Finally, Section 9 concludes this paper.

The codes associated with this article are publicly available at https://github.com/duowuyms/MANSY_ImmersiveVideoStreaming.

2 Motivation and Analysis

2.1 Impact of Viewing Pattern Diversity

The panoramic nature of immersive videos allows users to freely rotate their heads to watch the most attractive parts of the videos. As different users often have different viewing preferences, users' viewing patterns naturally exhibit high diversity [12] [13], posing unique challenge to the design of viewport prediction model.

Conventional approaches [7] [10] simply trains time-series NN models for viewport prediction with a set of

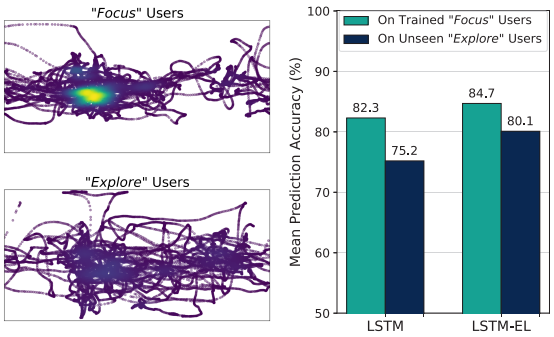


Fig. 1. Viewport prediction accuracy on two set of users with different viewing patterns.

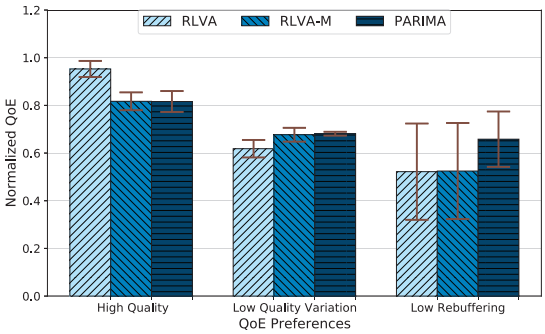


Fig. 2. Performance of bitrate selection methods on three different QoE preferences.

collected users' viewport samples, but neglect the model prediction bias towards the training dataset. When users' actual viewing patterns differ significantly from those observed in the training dataset, their models may suffer from significant accuracy loss, resulting in poor generalization. We implement an LSTM model [10] to demonstrate the ineffectiveness of these approaches. We set the historical and prediction window to be 1 second, and use 8x8 as the tiling scheme. We consider prediction accuracy as the performance metric, which is calculated as the intersection of union of the predicted and ground-truth viewports [7]. Besides, two set of users' viewports are sampled from a open dataset [24]: the *Focus* users prefer to focus on the objects in the video center, while the *Explore* users prefer to explore the entire scene, as depicted in Figure 1. We use the *Focus* set for training and consider the *Explore* set as users with unseen viewing patterns. The measurement results are presented in Figure 1. As depicted, *LSTM* achieves a high prediction accuracy of 82.3% on the trained *Focus* users. However, when confronted with users exhibiting significantly different viewing patterns, it struggles to accurately predict their viewport movements due to the inherent bias towards the training dataset. This leads to a noticeable performance drop on unseen *Explore* users, with an absolute accuracy loss up to 7.2%.

One way to combat the above limitation is to train the model over a large diverse dataset, but this requires tremendous amount of data with rich statistical diversity [13], which, however, is practically unavailable. Grouping users based on their viewing patterns and training personalized models for each group seems plausible [13] [17], but this necessitates retraining of a new model whenever a new group emerges, leading to prohibitive training cost. Previous work [8] has showcased the potential of ensemble learning (EL) to reduce prediction bias and improve model generalization. It works by training multiple independent sub-models and combining their predictions to yield more accurate results. The rationale behind is that when confronted with unknown data samples, the ensemble members may exhibit bias towards different directions, but their ensembled predictions can yield well-calibrated results that offset the bias [20] [25]. For illustrative purpose, we also implement an ensembled-version of LSTM model (denoted as *LSTM-EL*) following the approach in [8], which includes three independent LSTM sub-models. As shown in Figure 1, *LSTM-EL* is more robust to unseen viewing patterns: it maintains a high prediction accuracy of 80.2% on unseen *Explore* users with a smaller accuracy drop (4.2%) compared to *LSTM* (7.2%).

Despite promising, the challenge of EL is the substantial computation overhead caused by explicit model duplication, hindering its deployment on resource-constrained client devices. For instance, compared with *LSTM*, *LSTM-EL* introduce an extra overhead of 200% in terms of model parameters and floating-point operations. To tackle this challenge, we design an efficient Multi-viewport Trajectory Input Output (MTIO) architecture based on implicit EL. The MTIO architecture implicitly trains multiple sub-models without model duplication by establishing multiple input-output heads, thus exploiting the benefit of EL with negligible overhead. Moreover, we design the model architecture based on Transformer, which further improves the predictive performance of our model.

2.2 Impacts of QoE Preference Diversity

During the streaming of immersive videos, users' QoE is characterized by various metrics, such as video quality, rebuffering time, and quality variation [26] [27]. To effectively optimize multiple metrics, one common practice is to define a linear function that assigns different weights to each metric. Since users often have different QoE preferences (e.g., prioritizing high quality or low rebuffering), the assigned weights can vary among users [14] [16]. However, previous DRL methods [10] [11] [4] consider a fixed-weight QoE function as the reward to train the bitrate selection model, which limits their ability to generalize across diverse QoE preferences. Consequently, the discrepancy between users' actual QoE preferences and the optimized one can significantly degrade their performance [16], resulting in a poor video watching experience for users. Existing works [16] [18] have proposed training the DRL model with multiple QoE functions simultaneously to address this challenge. However, this naive approach suffers from the catastrophic forgetting problem [19], where the knowledge learned to optimize previous QoE preferences is overwritten by knowledge learned for current ones. Moreover, it may still experience performance degradation when serving users with unseen QoE preferences that differ from those optimized during training.

To illustrate the limitation of existing DRL-based solutions, we take *RLVA* [4], a state-of-the-art DRL bitrate selection model that optimizes a single QoE preference, as an

example. We train *RLVA* on the *High Quality* QoE preference and regard *Low Quality Variation* and *Low Rebuffering* as unseen preferences (detailed descriptions of QoE preferences are provided in Section 5). For comparative analysis, we also implement a heuristic approach *PARIMA* [28] and report its performance across the three preferences, as shown in Figure 2.

As expected, *RLVA* exhibits superior performance to *PARIMA* on the trained *High Quality* preference, with an average QoE increase of 16.7%. However, its performance significantly deteriorates on the unseen preferences, resulting in lower QoE compared to *PARIMA*. The poor performance of *RLVA* is attributed to its utilization of a fixed QoE function, which fails to guide the model in selecting appropriate bitrates for different preferences. Next, we further train *RLVA* using both *High Quality* and *Low Quality Variation* preferences (referred to as *RLVA-M*) to demonstrate the ineffectiveness of naive training with multiple QoE functions. As depicted in Figure 2, while *RLVA-M* achieves comparable performance to *PARIMA* on the trained *Low Quality Variation* preference, it experiences performance degradation on the trained *High Quality* preference compared to *RLVA*. This degradation stems from the forgetting problem, as the knowledge acquired to optimize *High Quality* is overwritten by the knowledge to optimize *Low Quality Variation*. Additionally, when serving users with the unseen *Low Rebuffering* preference, *RLVA-M* still performs worse than *PARIMA*, achieving poor generalization.

The key to optimize diverse QoE preferences is to learn useful representations about the relationship between QoE and the selected bitrates. Hence, in this paper, we leverage the advanced representation laerning (RepL) [23] technique to tackle the QoE preference diversity challenge. Specifically, we augment the reward function for model training with mutual information, which encourages the model to learn hidden representations that expose salient attributes of users' QoE preferences. The learned useful representations empowers our model to dynamically select bitrates based on users' preferences and generalize to diverse preferences including those unseen during the training stage.

3 System Overview

Figure 3 depicts the system overview of MANSY, which comprises two core components: MTIO-Transformer viewport prediction and RepL-based bitrate selection.

MTIO-Transformer viewport prediction. On the client side, a user watches an immersive video with a head-mounted device (HMD) such as Facebook Oculus and Microsoft Hololens, which continuously records the user's viewport trajectory. The historical viewport trajectory is extracted from the HMD and passed to the multiple input heads of the MTIO-Transformer. Based on the received trajectory, the model outputs multiple trajectories and ensembles them to generate an accurate prediction with small bias. The ensembled trajectory is subsequently passed to the bitrate selection module as an important reference for determining tile bitrates.

RepL-based bitrate selection. Given the predicted viewports as well as user's QoE preference information, a DRL agent parameterized by an NN model is used to dynamically select tile bitrates based on the environment state (e.g., network bandwidth conditions and playback buffer size). Once the agent makes the bitrate decisions, the HMD then requests tiles at the corresponding bitrates, downloads tiles from the server and displays them to the user. Note that the agent is offline trained with the RepL technique, where a QoE identifier is designed to facilitate the agent to disentangle highly semantic and useful representations from the QoE preference information. The QoE identifier is essentially another NN model used to guide the agent to maximize the mutual information between agent's selected bitrates and user's QoE preference.

4 MTIO-Transformer Viewport Prediction

Figure 4 depicts the architecture of the viewport prediction model of MANSY, which incorporates two core designs:

- *MTIO architecture.* We design the model with a Multi-viewport Trajectory Input Output (MTIO) architecture to efficiently reduce the prediction bias, so that our model can generalize across a broad range of users with diverse viewing patterns. Our model utilizes the insight that a neural network is over-parameterized and has sufficient capacity to fit multiple sub-networks simultaneously [21]. It therefore trains multiple independent sub-models within one network by establishing multiple input-output heads, with each head implicitly representing a sub-model. While each sub-model (i.e., head) may exhibit bias, the ensemble of their predictions can result in well-calibrated outcomes that effectively reduce such bias [20] [25], thus leading to improved generalization. Moreover, since sub-models are trained without explicit duplication, our model can utilize the benefits of ensemble with negligible overhead.
- *Transformer-based backbone.* Considering that long-term dependencies greatly influence time series prediction tasks including viewport prediction, we further design the backbone of our model based on Transformer. It leverages the attention mechanism [22] to effectively learn long-term dependencies to predict the trends of viewport movements more accurately, thus further improving the predictive performance.

4.1 Model Design

Let $\mathbf{v}_t = (x_t, y_t)$ denote the user's viewport at timestep t, where x_t, y_t represent the horizontal and vertical coordinates of viewport center in the equirectangular projection of the video[1], respectively. The detailed design of the model is explained as follows.

Multi-head inputs. As shown in Figure 4, our model incorporates M input heads. At any timestep t during video playback, it takes M historical viewport trajectories $\{\hat{\mathbf{v}}^i_{t-h}, \cdots, \hat{\mathbf{v}}^i_t\}^M_{i=1}$ as inputs, where h represents the historical time horizon. To ensure each head is independently trained, during the training stage, the parameters of each

1. The concepts in this section can be easily extended to other forms of coordinates, such as Euler angles and quaternion.

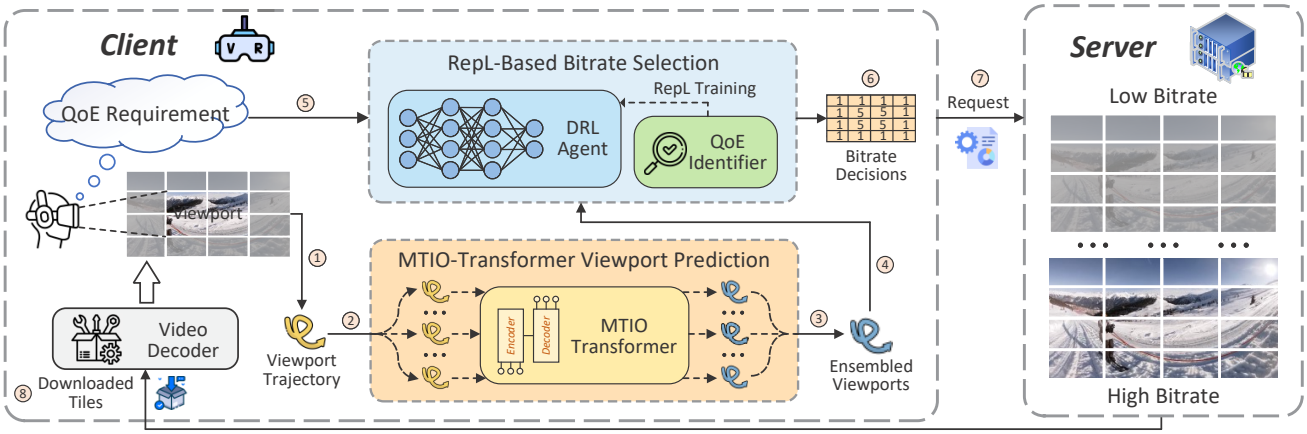


Fig. 3. System framework of the proposed tile-based immersive video streaming system MANSY.

head are randomly initialized and the M historical trajectories are randomly sampled from the training dataset. All trajectories are stacked, projected into a sequence of d_e-dimension embeddings and passed to the backbone network to extract underlying features.

Encoder-decoder. The backbone network adopts an encoder-decoder architecture, as illustrated in Figure 4. Both the encoder and decoder consist of N_{block} stacked blocks to extract complex features. The core of each block is the attention mechanism which enables effectively learning of long-term dependency information from the input embeddings. The attention mechanism can be described as mapping queries and sets of key-value pairs to attention weights that are assigned to different elements of the input sequence [22]. Specifically, let $Q \in \mathbb{R}^{d_k \times d_e}, K \in \mathbb{R}^{d_k \times d_e}, V \in \mathbb{R}^{d_v \times d_e}$ represent the query, key and value matrix, respectively. The attention weights are computed by:

$$Attention(Q, K, V) = \text{softmax}(QK^T/\sqrt{d_k})V \quad (1)$$

where $\sqrt{d_k}$ is the scale factor. The attention mechanism can be repeated by N_{ah} heads with different subspaces of Q, K, V, which benefits the model to jointly consider information from different representation subspaces. The multi-head attention is achieved by:

$$\begin{aligned} MultiHead(Q, K, V) &= \text{concat}([attn\_head_j]_{j=1}^{N_{ah}})W^O, \\ attn\_head_j &= Attention(QW_j^Q, KW_j^K, VW_j^V) \end{aligned} \quad (2)$$

where $W_j^Q \in \mathbb{R}^{d_e \times d_k}, W_j^K \in \mathbb{R}^{d_e \times d_k}, W_j^V \in \mathbb{R}^{d_e \times d_v}$ and $W^O \in \mathbb{R}^{N_{ah} d_v \times d_e}$ are all learnable weight matrices.

The encoder generates a sequence of hidden features extracted from the historical viewports, which, however, may contain redundant information [29]. To address this issue, rather than directly feeding the entire sequence of features to the decoder, we further design a distillation module to prioritize the dominant features from the encoder outputs. The distillation module comprises a 1D convolution layer and a max-pooling layer, as shown in Figure 4. It is used to compress the sequence length of the encoder outputs, resulting in a more focused set of input features for the decoder. This enhancement enables the decoder to better capture the viewport moving patterns. Additionally, another benefit of the distillation process is the reduction

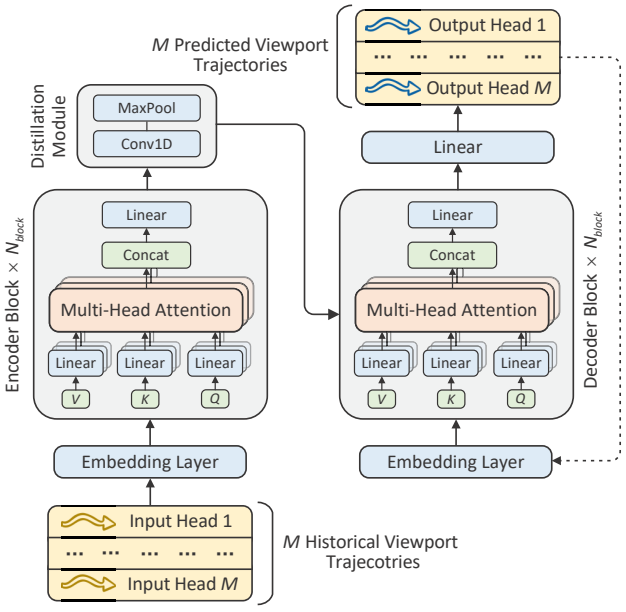


Fig. 4. The architecture of the proposed MTIO-Transformer viewport prediction model.

in computational workload for the decoder, as the smaller input feature length requires less computation.

Multi-head outputs. The model generates the future predicted viewports by linearly projecting the outputs of the decoder. In particular, our model is designed with M output heads. Each output head produces a viewport trajectory that corresponds to the prediction result of the corresponding input head. In addition, our model predicts M viewport trajectories in the autoregressive manner. It progressively predicts the viewports of next timestep by repeatedly injecting the previous predictions as inputs of the decoder. As a result, the future M viewport trajectories $\{\mathbf{v}_{t+1}^i, \cdots, \mathbf{v}_{t+H}^i\}_{i=1}^M$ are produced by the model, where H denotes the prediction horizon.

Loss function. The model is trained to minimize the distance between its predicted viewports $\{\mathbf{v}_{t+1}^i, \cdots, \mathbf{v}_{t+H}^i\}_{i=1}^M$ and ground truth viewports $\{\hat{\mathbf{v}}_{t+1}^i, \cdots, \hat{\mathbf{v}}_{t+H}^i\}_{i=1}^M$. To accurately measure the distance between two viewports, we design a distance function based on mean square error, which considers the periodicity of viewport horizontal coor-

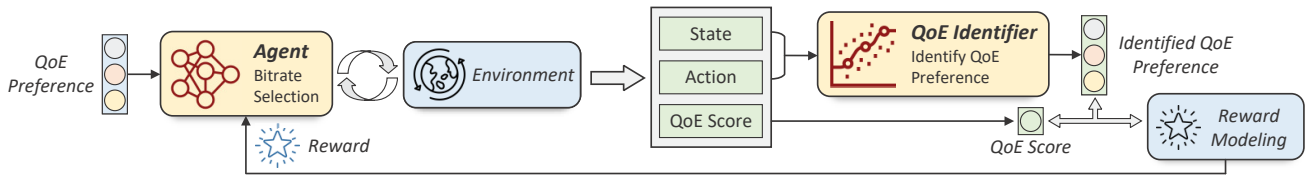


Fig. 5. Illustrations of the proposed RepL-based learning framework. Note that the reward for training the agent is composed of two parts, which are derived from the QoE score and outputs of QoE identifier.

dinate [28]. Specifically, let $\mathbf{v} = (x, y), \hat{\mathbf{v}} = (\hat{x}, \hat{y})$ represent the predicted and ground truth viewports, respectively, their distance is calculated by:

$$\begin{aligned} Dist(\mathbf{v}, \hat{\mathbf{v}}) &= \big(Dist_x(x, \hat{x})^2 + Dist_y(y, \hat{y})^2\big)/2 \\ Dist_x(x, \hat{x}) &= \min(|x - \hat{x}|, |x + w - \hat{x}|, |x - w - \hat{x}|) \\ Dist_y(y, \hat{y}) &= |y - \hat{y}| \end{aligned} \tag{3}$$

where w are the width and height of the video, respectively. The loss function is then defined as the sum of distances of each input-output head:

$$Loss = \sum_{i=1}^{M} Dist(\mathbf{v^i}, \hat{\mathbf{v}}^{\mathbf{i}}) \tag{4}$$

4.2 Ensembling Predictions

During the inference phase, the model extracts a historical viewport trajectory $\{\hat{\mathbf{v}}'_{t-h}, \cdots, \hat{\mathbf{v}}'_t\}$ from the HMD to predict the future viewports. This trajectory is duplicated by M times and fed to each input head, i.e., $\{\hat{\mathbf{v}}^i_{t-h}, \cdots, \hat{\mathbf{v}}^i_t\}_{i=1}^M = \{\hat{\mathbf{v}}'_{t-h}, \cdots, \hat{\mathbf{v}}'_t\}$. The output heads will produce M independent prediction results for the same input. Their predictions are finally ensembled to yield calibrated viewports that effectively reduce the prediction bias to improve the predictive performance:

$$\mathbf{v}'^e_{t+j} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{v}'^i_{t+j}, \forall j \in \{1, \cdots, H\} \tag{5}$$

where $\mathbf{v}'^e_{t+j}$ represents the ensembled predicted viewport.

Note that having M input-output heads only introduces negligible computation overhead. This is because the MTIO architecture only requires additional model parameters in the input and output layers, and meanwhile the model can obtain well-calibrated predictions in just a single forward pass. The detailed analysis of computation overhead of MTIO is covered in Section 6.2.

5 RepL-based Bitrate Selection

In this section, we present the detailed design of the bitrate selection approach of MANSY. Figure 5 illustrates the proposed RepL-based learning framework, which compromises a DRL agent for bitrate selection and a QoE identifier model to facilitate the agent to learn useful representations of users' QoE preferences. Under the current state of the environment, the agent generates a bitrate action and receives a QoE score as a partial reward signal. Notably, the outputs of the QoE identifier also serves as a reward signal, which captures the mutual information between users' QoE preferences and agent's selected bitrates. The final reward to train the agent is then derived from the combination of QoE score and outputs of QoE identifier.

In the following, we begin by describing the QoE model that characterizes the users' QoE preferences, then we elaborate the details of the proposed RepL-based bitrate selection algorithm, including DRL agent design, mutual-information-based reward modeling, as well as training methodology.

5.1 Quality of Experience Model

Following previous works [11] [13], we adopt three critical metrics to characterize the user's QoE, namely average viewport quality, quality variation and rebuffering time. These metrics are defined as follows:

1) Average viewport quality. The average viewport quality QoE^1_c describes the average bitrate quality of tiles inside user's actual viewport of chunk c. This metric can be calculated by:

$$QoE^1_c = \frac{\sum_{i=1}^{N_{tile}} \hat{v}_{c,i} r_{c,i}}{\sum_{i=1}^{N_{tile}} \hat{v}_{c,i}} \tag{6}$$

where N_{tile} is the total number of tiles; $\hat{v}_{c,i}$ is a boolean variable indicating whether i-th tile of chunk c is inside user's actual viewport; $r_{c,i}$ stands for the bitrate allocated to i-th tile.

2) Quality variation. The variations of tile quality inside the viewport and the viewport quality between two consecutive chunks should be smooth to avoid causing sickness or headache to users. The viewport quality variation QoE^2_c is measured by:

$$QoE^2_c = \frac{\sum_{i=1}^{N_{tile}} |\hat{v}_{c,i} r_{c,i} - QoE^1_c|}{\sum_{i=1}^{N_{tile}} \hat{v}_{c,i}} + |QoE^1_c - QoE^1_{c-1}| \tag{7}$$

where the first term denotes the intra-variation of tile quality inside the viewport, and the second term denotes the inter-variation of viewport quality between consecutive chunks.

3) Rebuffering time. If the playback buffer goes empty before a chunk is downloaded, the user will suffer from a *rebuffering* event. Following previous methods [10] [13], we calculate the rebuffering time QoE^3_c by:

$$QoE^3_c = (l_c - b_c)_+ \tag{8}$$

where l_c is the download time of chunk c; b_c denotes the buffer occupancy when the download request of chunk c is sent; $(\cdot)_+$ represents the function of $\max(\cdot, 0)$.

Based on the above metrics, user's QoE for c-th chunk can be modeled as:

$$QoE_c = \lambda_1 QoE^1_c - \lambda_2 QoE^2_c - \lambda_3 QoE^3_c \tag{9}$$

Here, $\lambda_1, \lambda_2, \lambda_3$ are non-negative weight parameters that measure the relative importance of different metrics and satisfy $\lambda_1 + \lambda_2 + \lambda_3 = 1$. Therefore, we use $w = (\lambda_1, \lambda_2, \lambda_3)$ to represent the user's QoE preference. Similar models have also been adopted in [11] [13] [18].

5.2 DRL Agent Design

The DRL agent is responsible for bitrate allocation based on the user's QoE preference and environment state information. The inputs, outputs, and NN architecture of the agent are designed as follows.

Inputs. When determining tile bitrates for chunk c, the agent takes a QoE preference w and information of environment state s_c as inputs. Formally, the environment state s_c is modeled as:

$$s_c = (Z_c, R_c, \vec{v}_c, \vec{g}_c, \vec{n}_c, \vec{q}_c^1, \vec{q}_c^2, \vec{q}_c^3, b_c)$$

Here, Z_c and R_c record the sizes and bitrate qualities of each tile at different bitrate versions, respectively. $\vec{v}_c$ is the binary vector that indicates whether a tile is inside the predicted viewport. $\vec{g}_c$ denotes the viewport prediction accuracy of past k chunks. $\vec{n}_c$ is the vector of past k measured network throughputs. $\vec{q}_c^1, \vec{q}_c^2, \vec{q}_c^3$ record the average viewport quaity, quality variation and rebuffering time of the past k chunks, respectively. Finally, b_c represents the buffer occupancy.

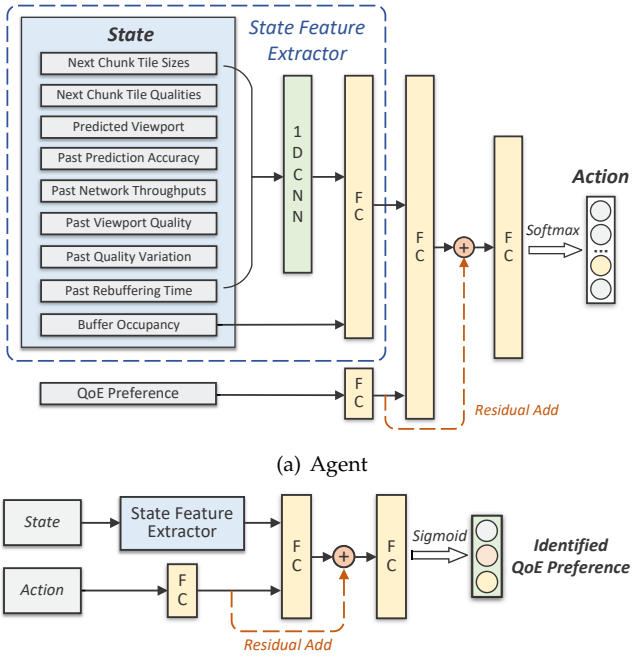
Outputs. Based on QoE preference w and state s_c, the agent outputs an action a_c that corresponds to the bitrates allocated to tiles inside and outside the predictied viewport. The action a_c is represented as $a_c = (r_c^{in}, r_c^{out})$, where $r_c^{in}, r_c^{out} \in \mathcal{R}$ and $\mathcal{R}$ denotes the discrete candidate bitrate set of tiles. In particular, r_c^{in}, r_c^{out} satisfy the constraint of $r_c^{in} \geq r_c^{out}$, as bitrate of tiles inside the viewport should be larger than that outside the viewport. All possible combinations of r_c^{in}, r_c^{out} constitute the discrete action space. Based on the output action, we employ a pyramid-based strategy to assign bitrates to tiles according to their distance to the predicted viewport. Specifically, we assign r_c^{in} to tiles inside the predicted viewport, then iteratively scale the boarder of viewport with one tile, assign $r_c^{out}/scale$ to the newly covered tiles[2] until all tiles are assigned with bitrates, where $scale$ denotes the scaling step. The rationale behind is that tiles distant from the predicted viewport are of lower viewing probability and therefore can be allocated with lower bitrates for bandwidth efficiency.

Network architecture. Figure 6(a) depicts the NN architecture of the agent. For vectorized information $Z_c, R_c, \vec{v}_c, \vec{g}_c, \vec{n}_c, \vec{q}_c^1, \vec{q}_c^2, \vec{q}_c^3$, we use 1D convolution layers to extract hidden features from each input. These features are next flattened and fed to fully-connected (FC) layers. The buffer occupancy b_c and QoE preference w are directly fed to FC layers for feature extraction. All hidden features are concatenated together and sequentially fed to another two FC layers to learn complex relationship between different features. Finally, a softmax layer is used to output the probability distribution of each action.

5.3 Mutual Information-Based Reward Modeling

The key to tackle the challenge of QoE preference diversity is to train the agent to understand the relationship between the input QoE preference and its output bitrate actions. One natural approach is to train the agent with multiple preferences with the reward defined as the QoE scores calculated with different preference weights. Nevertheless, as described in Section 2.2, this approach is prone to catastrophic forgetting and generalization issues. Moreover, even though the QoE preference is explicitly fed to the agent, without any restrictions on how such information should be utilized, the agent is free to ignore it for bitrate selection, which prevents the agent to maximize diverse QoE objectives according to the input preference.

2. If $r_c^{out}/scale \notin \mathcal{R}$, we replace the $r_c^{out}/scale$ with the closest bitrate version in $\mathcal{R}$ and bound it with $\min(\mathcal{R})$.



(a) Agent

(b) QoE identifier

Fig. 6. Neural network architectures of the agent and QoE identifier.

To combat the above limitations, we augment the reward for agent training based on representation learning (RepL). RepL is the advanced technique that enables effective learning of task-specific representations that reveal meaningful patterns and alleviate the complexity of the task-solving process [23]. Specifically, we introduce mutual information into the reward function to facilitate the agent to learn useful representations that capture the salient attributes of users' preferences (e.g., preferences of bitrate quality and rebuffering). In information theory, the mutual information $I(X;Y)$ between variable X and variable Y measures the amount of information learned from Y about X [30]. In our case, we expect to maximize the amount of information learned from the agent's output action a about the input preference weight w, so that the agent can dynamically select bitrates according to the input preference. In other words, the information of w should not be lost in the decision making process. Hence, the reward function to train the agent is defined as follows:

$$rew_c = (1-\alpha)QoE_c(w) + \alpha I(w; a_c, s_c) \qquad (10)$$

Here, rew_c stands for the reward signal of downloading chunk c. $QoE_c(w)$ represents the QoE scores calculated with preference weight w according to equation (9). $I(w; a_c, s_c)$ denotes the mutual information between the input preference w and agent's output action a_c selected when encountering state s_c. Finally, $\alpha \in [0, 1]$ is the weight parameter to control the trade-off between the two components.

Algorithm 1 Training procedure of RepL-based bitrate selection algorithm

Input: QoE preference pool $\mathcal{W}$; weight parameter α.
Output: Learned agent π_θ.

1: Initialize the parameters of agent as π_{θ_0}.
2: Initialize the parameters of QoE identifier as Q_{δ_0}.
3: **for** $i = 0, 1, 2, \cdots$ **do**
4: Sample a batch of QoE preferences: $w_i \sim \mathcal{W}$.
5: Sample trajectories: $\tau_i \sim \pi_{\theta_i}(w_i)$, with the preference weight fixed during each rollout.
6: Update $\delta_i \rightarrow \delta_{i+1}$ by descending with gradients:

$$\Delta_{\delta_i} = \mathbb{E}_{(s,a)\sim\tau_i}[\nabla_{\delta_i} MSE(w_i; Q_{\delta_i}(s,a))]$$

7: Take a policy update step from θ_i to θ_{i+1} using the PPO update rule to optimize reward:

$$rew = (1-\alpha)QoE(w_i) - \alpha \log MSE(w_i; Q_{\delta_{i+1}}(s,a))$$

8: **end for**

The technical challenge of implementing mutual information lies in the difficulty of exact computation [31]. Fortunately, recent advances have showcased the effective estimation of mutual information using NN models [30] [32] [33]. Therefore, we design QoE identifier, an efficient model to estimate mutual information. As demonstrated in Figure 5, the QoE identifier is essentially a neural regressor that identifiers the information of preference weight w from the action a_c that agent takes when encountering state s_c. It takes a state-action pair from the agent as input and outputs the identified QoE preference, which will be used for efficient estimation of mutual information. With the QoE identifier, the reward for training the agent is modeled as:

$$rew_c = (1-\alpha)QoE_c(w) - \alpha \log MSE(w; Q_\delta(s_c, a_c)) \quad (11)$$

where Q_δ represents the NN model of QoE identifier parameterized by δ, and $MSE(w; Q_\delta(s_c, a_c))$ denotes the mean square error between the true preference weight w and the one identified by QoE identifier $Q_\delta(s_c, a_c)$. The term $-\log MSE(w; Q_\delta(s_c, a_c))$ quantifies the amount of mutual information between the QoE preference and the selected action. The higher of its value, the greater similarity between true preference w and the identified one $Q_\delta(s_c, a_c)$, thereby indicating higher mutual information. The rationale behind is that if the agent's bitrate decisions capture the QoE preference w properly, the QoE identifier should be able to extract the information of w from its state s_c and action a_c, and vice verse. Therefore, through reward optimization, our agent is capable to adaptively maximize diverse QoE objectives based on the users' preferences.

Figure 6(b) illsutrates the architecture of the QoE identifier model. We design the model with the same state feature extractor as the agent. Besides, we process the action one-hot vector with a FC layer, pass the features sequentially to another two FC layers, and use sigmoid as the output layer. To be consistent to the number of QoE preference weight parameters, the number of output neurons is set to 3. It is worth noting that the QoE identifier is designed to guide the training of agent, and therefore is used only in the training phase.

5.4 Training Methodology

We introduce adversarial training to update the parameters of agent and QoE identifier. Let π_θ and Q_δ represent the agent parameterized by θ and QoE identifier parameterized by δ, respectively. At each training step, we sample a batch of preference weights from the weight pool: $w_i \sim \mathcal{W}$. Then we sample the state-action trajectories under the agent policy with the preference weight fixed during each rollout: $\tau_i \sim \pi_{\theta_i}(w_i)$. Next, we update the QoE identifier model in order to calculate $Q_\delta(s,a)$ in equation (11) properly. The goal of QoE identifier is to optimize the mutual information between the agent's state-action trajectory and the associated QoE preference to be maximum, which is equivalent to minimize the mean square error between the true QoE preference and its identified one. Hence, its parameters can be updated $\delta_i \rightarrow \delta_{i+1}$ through the following gradients:

$$\Delta_{\delta_i} = \mathbb{E}_{(s,a)\sim\tau_i}[\nabla_{\delta_i} MSE(w_i; Q_{\delta_i}(s,a))] \quad (12)$$

Afterwards, we update the agent $\theta_i \rightarrow \theta_{i+1}$ using common reinforcement learning (RL) framework to optimize the reward calculated by (11). In this paper, we choose Proximal Policy Gradient (PPO) [34] as the RL framework to update the agent's parameters. The overall training procedure is summarized in Algorithm 1.

Residual learning enhancement. During the practical implementation of the proposed RepL-based bitrate selection algorithm, we find that both the agent and QoE identifier face convergence difficulties (see Section 6.3). Empirically, we observe that when fusing QoE preference and state features in the agent model, the dominance of state features hinders the propagation of preference features within the network. A similar phenomenon occurs with the action features in the QoE identifier model. Consequently, the agent struggles to capture QoE preference information, while the QoE identifier fails to effectively identify the preference information from the agent's actions. To address this issue, we employ residual learning [35] to facilitate the propagation of these crucial features within the networks. Specifically, as depicted in Figure 6, we add the preference features and action features with the outputs of the penultimate FC layer for the agent and QoE identifier, respectively.

6 Evaluation

6.1 Experiment Setup

Datasets. We consider two large-scale immersive video datasets with viewport movement traces for evaluation, which contains numerous videos of various types watched by a large number of users:

- *Wu2017* [24]: In this dataset, 8 videos with an average length of 242 seconds are used for evaluation, with 6 for training, 1 for validation and 1 for testing. Each video in the dataset contains 48 viewport trajectories of 48 users.
- *Jin2022* [12]: We employ 24 60-second videos watched by 60 users from this dataset. We select 18 videos for training, 3 for validation and 3 for testing.

The viewport positions of these datasets are transformed into equirectangular format according to the method in [36]. Each video is segmented into chunks of 1 second, and each chunk is further divided into 8x8 tiles. We use FFMPEG

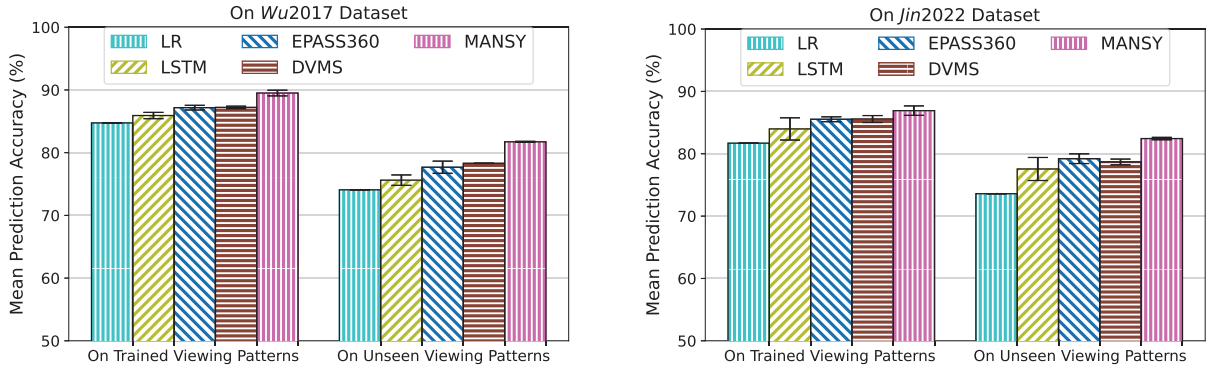


Fig. 7. Mean prediction accuracy of different methods on trained and unseen viewing patterns. We show the mean and standard deviation across 3 runs with different random seeds for training and testing.

with encoder X.264 to encode the videos into $|\mathcal{R}| = 5$ bitrate versions: 1Mbps (360p), 5Mbps (720p), 8Mbps(1080p), 16Mbps(2K), 35Mbps (4K). For bandwidth dataset, we consider a public dataset [37] to simulate real-world network conditions, which contains 40 bandwidth traces with various fluctuation patterns. We select 24 traces for training, 8 for validaiton and 8 for testing.

Baselines. We compare our approach with other state-of-the-art methods. Specifically, for viewport prediction, we implement the following baselines for comparison:

- *LR* [38] employs a simple linear regression model to predict the trends of viewport movements.
- *LSTM* [10] trains a basic LSTM model for viewport prediction.
- *EPASS360* [8] is similar to *LSTM*, except that it leverages model ensemble to explicitly set up three LSTM models and ensembles their prediction results.
- *DVMS* [6] designs a Gate Recurrent Unit (GRU) model based on variational auto-encoder (VAE), which predicts five viewport trajectories to capture the variation of users' viewing patterns at the cost of multiple forward passes during inference phase.

For bitrate selection, the following baselines are implemented:

- *PARIMA* [28] employs a heuristic algorithm that allocates bitrates to tiles according to their distance to the predicted viewports based on the estimated bandwidth.
- *RLVA* [11] designs a DRL algorithm for bitrate selection. It trains the DRL agent to optimize only one single QoE preference.
- *PAAS* [18] proposes a dynamic-preference scheme to train a DRL agent to simultaneously optimize multiple QoE preferences. It defines reward as the combination of QoE scores calculated under the current preference and the ones calculated under another randomly sampled preference, so as to alleviate forgetting problem.
- *Pensieve* [39] is a DRL-based bitrate selection algorithm for traditional non-immersive videos, which allocates the same bitrate to all tiles for each video chunk. We implement this method to verify whether the non-immersive video streaming algorithm can work effectively in the case of immersive video streaming.

Parameter settings. By default, the number of input-output heads of the proposed MTIO Transformer model is set to $M = 3$. Besides, we configure $d_e = 512, N_{ah} = 8, d_k = d_v = 64, N_{block} = 2$ and the learning rate as 1e-4. For bitrate selection, we empirically feed past $k = 8$ sample information into the agent. The number of filters of all 1D CNN layers is 128, and the stride is 1. Their kernel sizes are set to the length of input vectors. The size of the FC layer to extract features from QoE preference is set to 128, while the sizes of the last two FC layers are set to 1280, 128, respectively. The same setting are applied on the QoE identifier model. In addition, we configure the learning rate of the agent as 5e-4, reward discount factor as 0.95, entropy coefficient as 0.02 and weight parameter α as 0.5. The learning rate of the QoE identifier is configured as 1e-4.

Metrics. We use the prediction accuracy and QoE scores as the evaluation metrics. In particular, the prediction accuracy is measured as the intersection of union (IoU) between the predicted viewport and ground-truth viewport [7]. To be more specific, the prediction accuracy is calculated by:

$$Accuracy = \frac{FoV(x^p, y^p) \cap FoV(x^g, y^g)}{FoV(x^p, y^p) \cup FoV(x^g, y^g)}$$

where (x^p, y^p) and (x^g, y^g) denote the positions of the predicted and ground-truth viewport centers, respectively; $FoV(x, y)$ denotes the field of view (FoV), i.e., viewport area, centered at position (x, y). According to the specifications of existing popular headsets [40], we configure the size of FoV to be 16% of the total video area.

Hardware settings. We conduct all experiments on a desktop computer equipped with an Intel(R) Core(TM) i7-12700 CPU and NVIDIA 3090 GPU. Note that the computing resources to run our experiments are highly redundant.

6.2 Viewport Prediction

In this part, we first evaluate the performance of MTIO-Transformer viewport prediction model of `MANSY`. By default, we use the viewports in the last second to predict the future viewports in the next second, i.e., $h = H = 1$s. Additionally, we leverage the method in [13] to categorize users into seven groups of diverse viewing patterns based on their viewing preferences (e.g., preferring to watch dynamic objects or focus on static objects). We select five groups for training and the rest for evaluating generalization performance. When evaluating each method on the trained/unseen viewing patterns, we report the their performance on the trained/unseen groups on the testing videos.

TABLE 1
Comparison of computation overhead of different architectures. "↑ x%" means the increase compared to standard architecture with one input-output head ($M = 1$).

Architecture	Memory Consumption (MB)	Inference Time (ms)
Standard ($M = 1$)	28.12	35.63
MTIO ($M = 3$)	28.13 (↑ 0.04%)	35.66 (↑ 0.78%)
MTIO ($M = 5$)	28.15 (↑ 0.11%)	35.85 (↑ 1.43%)
MTIO ($M = 10$)	28.19 (↑ 0.25%)	36.35 (↑ 1.56%)
VAE [6]	32.12 (↑ 10.67%)	153.54 (↑ 430.94%)
Explicit Ensemble [8]	84.35 (↑ 200.00%)	106.89 (↑ 200.00%)

6.2.1 Comparison With Baselines

Figure 7 compares the mean viewport prediction accuracy of different methods. As shown in Figure 7, MANSY outperforms other baselines on both trained and unseen viewing patterns. On *Wu2017* (*Jin2022*) dataset, it effectively improves the absolute mean accuracy by 2.3%–4.8% and 3.4%–7.7% (1.3%–5.2% and 3.8%–8.8%) on trained and unseen viewing patterns, respectively. Notably, MANSY shows more significant improvement on the unseen viewing patterns, which highlights its superior generalization performance. The superiority of MANSY can be attributed to two key aspects.

- First, MANSY employs the MTIO architecture to efficiently reduce the prediction bias and thus achieves better generalization performance over diverse viewing patterns. In contrast, *LSTM* neglects the prediction bias towards the training data and thus experiences more drastic accuracy loss when testing on unseen viewing patterns.
- Second, MANSY designs the prediction model based on Transformer with the attention mechanism to effectively learn long-term dependencies and predict the trends of viewport movement more accurately. By comparison, both *EPASS360* and *DVMS* adopt conventional LSTM or GRU models for viewport prediction, which limit their ability to capture the viewport moving patterns, thus resulting in poorer performance than MANSY.

We further compare the accuracy of different methods with different prediction horizon H on *Wu2017* dataset. As depicted in Figure 8, MANSY consistently achieves the highest prediction accuracy, with the improvements of 2.4%–17.8% and 4.2%–30.8% on trained and unseen viewing patterns, respectively. This indicates the stronger ability of MANSY in both short-term and long-term prediction. The results on *Jin2022* dataset are similar and thus are omitted here for brevity.

6.2.2 Effectiveness of MTIO Architecture

Next, we evaluate the effectiveness of the proposed MTIO architecture. We report the mean accuracy of MANSY with different number of input-output heads M in Figure 9. Note that $M = 1$ is equivalent to the standard architecture with single input-output head. As shown, increasing M will improve the predictive performance especially on unseen viewing patterns, which confirms the effectiveness of MTIO architecture to reduce prediction bias and improve generalization. Besides, we also observe that such performance gain will gradually diminish when M is sufficiently large (e.g., $M > 3$). This phenomenon could be attributed to the fact that our MTIO architecture utilizes a single neural network to implicitly train multiple sub-models, and a large M will quickly reach the network capacity. Consequently, the trained sub-models may share high similarities, while the success of ensemble learning relies on the diversity of sub-models [21]. As a result, their prediction bias may accumulate, hurting the benefits of ensemble. This suggests that in practice, it is unnecessary to set M too large for the MTIO architecture.

As a supplement, we also measure the computation overhead of MTIO architecture in terms of the increase of the memory consumption and inference time[3] compared to the standard architecture (i.e., $M = 1$). As shown in Table 1, the overhead introduced by MTIO is negligible: even when $M = 10$, it only increases 0.25% of memory consumption and 1.56% of inference time. By comparison, when adopting VAE or explicit ensemble, as in [6] [8], the overhead drastically increase to 10.67%/430.94% or 200%/200% on memory consumption and inference time, respectively.

We also measure the overhead of the *LR* method and results show that it consumes 2.66 MB memory and takes 11.77 ms per inference. Considering that *LR* is a non-DNN-based method, it naturally causes less computation overhead than DNN-based ones, including our MTIO Transformer. Despite this, our MTIO Transformer achieves significantly higher prediction accuracy than *LR*, especially for long-term prediction, as illustrated in Figure 8. Moreover, modern commercial VR headsets are now equipped with sufficient computing resources to support the exeuction of small-size DNNs. For example, Meta Quest Pro [41] is powered by the Qualcomm Snapdragon XR2@1.8GHz, which features 8 CPU cores and 12GB memory along with an Adreno 650 GPU for accelerating computation. Hence, deploying our model on commercial headsets is feasible and the performance-overhead trade-off of our model is worthwhile.

6.3 Bitrate Selection

In this part, we evaluate the performance of MANSY in bitrate selection. Following previous work [18], we construct 8 QoE weights with diverse preferences[4] on different QoE metrics (e.g., high-bitrate-first and low-rebuffering-first), with 4 used for training and the rest for generalization evaluation. In the following, we report the performance of different methods on the trained/unseen QoE preferences on the testing videos and users. Besides, since *RLVA* follows the single-preference optimization scheme, we follow the idea in [10] [13] [17] to train personalized *RLVA* models for each QoE preference separately. When it comes to an unseen preference, we use the *RLVA* model trained with the QoE preference of the highest cosine similarity for testing. We adopt the same strategy for *Pensieve*, the single-preference algorithm originally designed for non-immersive video streaming.

3. Time measurement is performed on an Intel(R) Core(TM) i7-12700 CPU, and is limited to *use only 1 CPU core* to simulate resource-constrained scenarios.

4. The full list of the QoE preferences is omitted here for brevity and can be found in our codes.

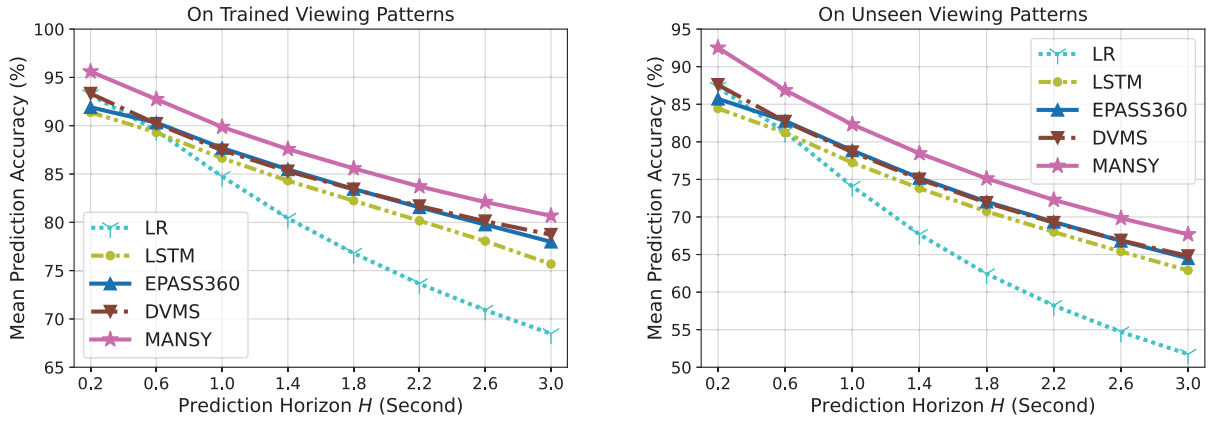


Fig. 8. Mean prediction accuracy of different methods with different prediction horizon H on *Wu2017* dataset.

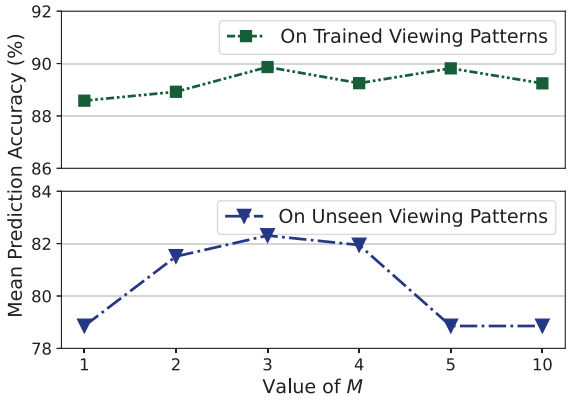


Fig. 9. Effects of the number of input-output heads M.

6.3.1 Comparison With Baselines

Figure 10 compares the performance of different methods on different sets of preferences and datasets in terms of mean QoE and QoE distribution. As shown, MANSY outperforms other methods in different cases. Specifically, on *Wu2017* (*Jin2022*) dataset, compared to *Pensieve*, *PARIMA*, *RLVA* and *PAAS*, MANSY improves the average QoE by 14.1%, 9.2%, 7.4%, 3.1% and 15.3%, 9.3%, 9.7%, 5.1% (6.7%, 6.0%, 4.3%, 3.5% and 13.2%, 10.2%, 8.9%, 3.4%) on trained and unseen QoE preferences, respectively. In particular, MANSY generally demonstrates more significant improvement on unseen preferences, thus achieving stronger generalization ability. Besides, a large proportion of QoE values of MANSY is concentrated in the larger range, which further demonstrates the superiority of MANSY. The performance gain of MANSY stems from the design of QoE identifier based on RepL to train the agent to maximize the mutual information between QoE preferences and bitrate decisions, enabling the agent to generalize across diverse QoE preferences.

To gain a comprehensive understanding of the performance of each method, Figure 11 presents the mean values of different QoE metrics of different methods across various preferences. From Figure 11, we can see that *Pensieve* consistently yields low quality variation across all cases and matches MANSY's performance for preferences that emphasize low quality variation (e.g., Figure 11(d)). This is because *Pensieve* allocates the same bitrates for all tiles of the video chunks and thus removes intra-variation of viewport quality. However, such strategy will allocate unnecessarily high bitrates to tiles outside viewports. Therefore, *Pensieve* consistently yields low viewport quality or high rebuffering time (e.g., Figure 11(b) and Figure 11(c)), as tiles outside viewports could be allocated with lower bitrates to reserve bandwidth for improving the tile quality inside viewports or reducing buffering time. In consequence, *Pensieve* generally achieves lower QoE than other immersive video streaming algorithms in most cases, as depicted in both Figure 10 and Figure 11. This indicates that the non-immersive video streaming algorithm is not well suited for immersive videos

As a heuristic algorithm, *PARIMA* allocates bitrates to tiles according to the estimated bandwidth. However, due to the underestimation of bandwidth, *PARIMA* tends to conservatively select low bitrates to prevent rebuffering. Consequently, it achieves comparable performance to MANSY only on preferences that emphasize low rebuffering (e.g., Figure 11(b)), while significantly underperforming in other cases. For *RLVA*, although it may achieve satisfactory performance on the unseen preference similar to the one optimized during training (e.g., Figure 11(a) and Figure 11(c)), it fails to accurately capture user's QoE preference for bitrate selection. For instance, as shown in Figure 11(c), *RLVA* trained to aggressively optimize viewport quality adopts a similar bitrate selection strategy for users with requirement on maintaining low variation. In consequence, it exhibits the largest quality variation, leading to inferior performance compared to MANSY.

Despite incorporating a dynamic-preference scheme to train the agent for optimizing different QoE preferences, we observe that *PAAS* still suffers from the forgetting problem. For example, while *PAAS* outperforms *RLVA* on the trained low-rebuffering-first preference (Figure 11(b)), it demonstrates inferior performance compared to *RLVA* on another trained high-quality-first preference (Figure 11(a)). Moreover, *PAAS* also exhibits limited generalization ability, as it may perform worse than *RLVA* in some cases even on the unseen preferences (e.g., Figure 11(c)). In contrast, thanks to the proposed RepL-based training scheme, MANSY successfully learns useful representations that capture the essential characteristics of users' QoE preferences, enabling

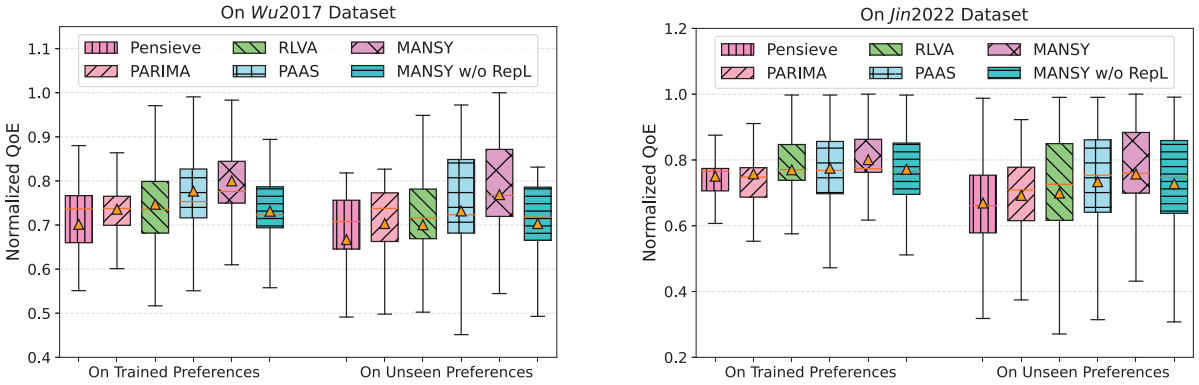
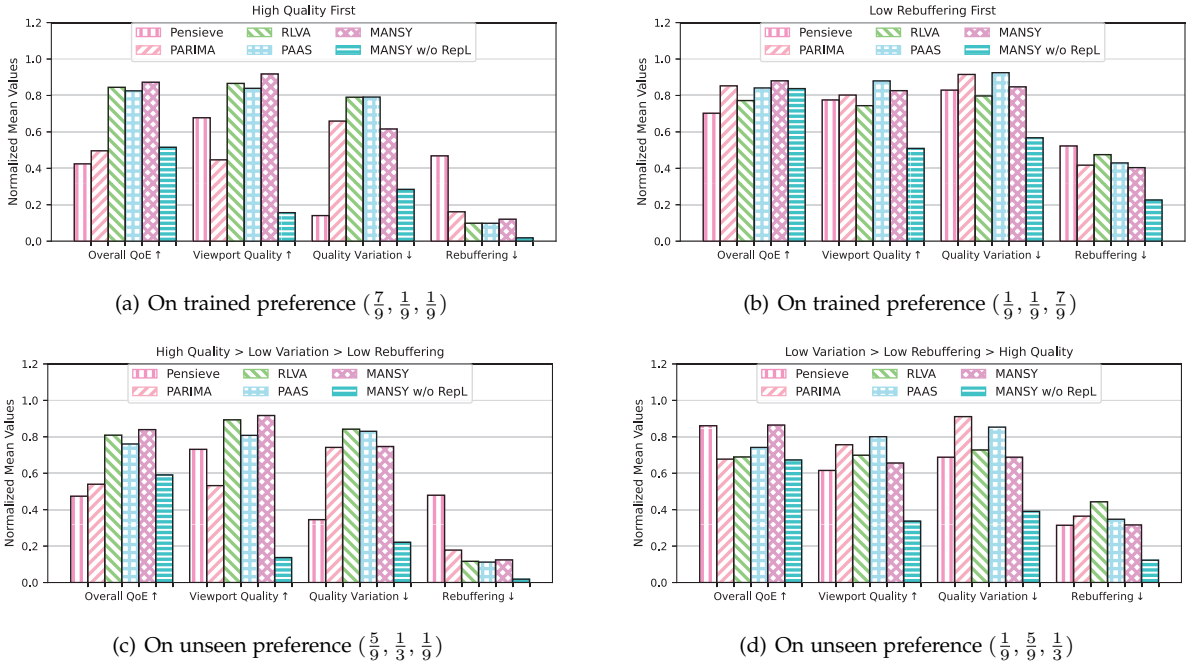


Fig. 10. Normalized QoE of different methods on trained and unseen QoE preferences. The shape of box shows the QoE distribution and the triangle in each box denotes the mean QoE.



(a) On trained preference $(\frac{7}{9}, \frac{1}{9}, \frac{1}{9})$

(b) On trained preference $(\frac{1}{9}, \frac{1}{9}, \frac{7}{9})$

(c) On unseen preference $(\frac{5}{9}, \frac{1}{3}, \frac{1}{9})$

(d) On unseen preference $(\frac{1}{9}, \frac{5}{9}, \frac{1}{3})$

Fig. 11. Normalized mean values of different QoE metrics of different methods on various preferences on *Wu2017* dataset. The arrow ↑ / ↓ means higher/lower is better.

dynamic bitrate selection based on input preferences. Notably, the learned representations are generalizable, empowering the agent to optimize unseen preferences (e.g., Figure 11(d)). As a result, MANSY demonstrates the most promising performance across all cases.

6.3.2 Ablation Study

In this part, we set up several experiments to provide a thorough understanding of the bitrate selection model of MANSY, including effectiveness of RepL-based training and residual learning.

Effectiveness of RepL. We remove the proposed RepL training scheme from MANSY to explore its contributions to the bitrate selection performance of MANSY. Specifically, we simply train the agent with multiple QoE preferences simultaneously but without the guidance of the QoE identifier, which forms the method of *MANSY w/o RepL*.

Figure 10 illustrates the performance of *MANSY w/o RepL* on both trained and unseen viewing patterns on each dataset. It can be seen that without RepL, the performance of *MANSY w/o RepL* significantly decreases and even becomes worse than baselines. To gain further insights, we analyze its performance on individual QoE preferences in Figure 11. As depicted in Figure 11, *MANSY w/o RepL* consistently maintains the lowest rebuffering time on all cases. This suggests that without the guidance of QoE identifier to learn informative representations of users' QoE preferences, *MANSY w/o RepL* heavily suffers from the *catastrophic forgetting* problem. The dominance of knowledge learned to optimize low rebuffering forces *MANSY w/o RepL* to aggressively select low bitrates regardless of users' actual QoE preferences. As a result, it only achieves better or competitive performance than baselines on the low-rebuffering preference (e.g., Figure 11 (b)), but achieves inferior performance on other ones (e.g., Figure 11 (c)). This finding

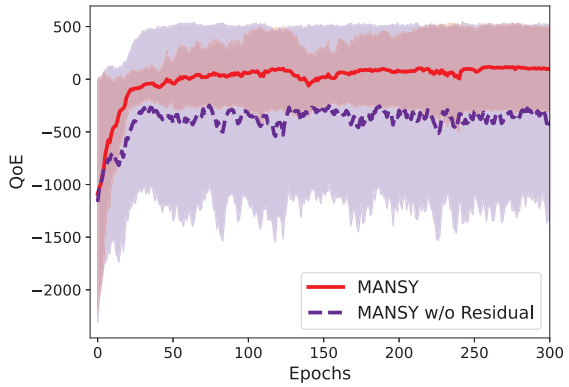


Fig. 12. Convergence comparison of MANSY with/without residual learning.

further supports our previous observations[5] in Section 2.2 that simply training with multiple QoE preferences will suffer from severe catastrophic forgetting issues and lead to poor generalization performance.

In contrast, MANSY significantly outperforms *MANSY w/o RepL* and other baselines across all cases. This outcome demonstrates that MANSY efficiently tackles the challenge of catastrophic forgetting and achieves strong generalization performance, which confirms the effectiveness of the proposed RepL-based training scheme.

Effectiveness of residual learning. We next compare the convergence performance of MANSY and *MANSY w/o Residual* to validate the effectiveness of residual learning. To be more specific, the NN architectures of the agent and QoE identifier of *MANSY w/o Residual* are designed without the residual layers depicted in Figure 6. As illustrated in Figure 12, *MANSY w/o Residual* faces issues related to training instability and poor asymptotic performance (i.e., final performance after convergence [42]). This is attributed to the fact that the important features of QoE preference are lost during the fusion with state features, which make *MANSY w/o Residual* difficult to capture the QoE preference information. In contrast, MANSY efficiently addresses this issue by adding residual layers into the NN models to facilitate the propagation of these critical features within the models. Hence, MANSY achieves better training stability and asymptotic performance.

7 Related Work

Viewport prediction. As one of the main building block of tile-based immersive video streaming, the design of viewport prediction models has been extensively studied [27] [43]. Considering the diversity of users' viewing patterns, recent studies have explored several approaches to enhance the viewport prediction models to resolve the diversity challenge. For instance, the works in [13] [17] group users with similar viewing patterns and train separate models for each group. The major limitation of this approach is that each time a new user group emerges, a model retraining process is required, resulting in prohibitive training cost. Alternatively, some researchers attempt to improve the generalization of viewport prediction models to serve a broad range of users. For instance, Guimard et al. [6] design a variational auto-encoder model that predicts multiple viewports to capture the variation of user's viewing patterns. Zhang et al. [8] explicitly train several LSTM models and ensemble their prediction results to yield calibrated predictions. These methods, however, require model duplication or multiple forward passes during inference phase, leading to significant computation cost. By comparison, MANSY designs an efficient MTIO-Transformer model based on implicit ensemble learning. It can obtain well-calibrated predicted viewports with a single model and a single forward pass, thus improving generalization with negligible overhead.

Bitrate selection. Recent years have witnessed the successful applications of deep reinforcement learning (DRL) in bitrate selection of tile-based streaming [11] [4] [14]. However, the performance of DRL-based methods is restricted in real-world conditions where users exhibit high diversity on QoE preferences. To tackle this challenge, previous works [10] [4] propose to train different bitrate selection agents with different QoE preferences, which, however, results in prohibitive training cost and lacks scalability to adapt to new preferences. Li et al. [14] designs a muli-agent DRL solution to allocate bitrates to users that watch the same video and share the same bottleneck link while considering their QoE preferences. Yet, their solution is built upon the strict assumption of fixed bandwidth of the bottleneck link, which often does not hold true in practice and thus limits its performance in real-world scenarios. On the other hand, Wu et al. [18] design a dynamic-preference scheme to train the DRL agent with multiple QoE preferences simultaneously. Nevertheless, this approach suffers from the catastrophic forgetting problem, failing to generalize across diverse QoE preferences. In contrast, MANSY leverages the advanced RepL technique to train the agent without any specific presumptions. It designs an efficient QoE identifier to encourage the agent to automatically extract useful knowledge from users' QoE preferences, thus generalizing across users with diverse preferences.

8 Discussion

In this section, we discuss the potential methodologies to further enhance the performance of MANSY. Regarding viewport prediction, the current implementation of the MTIO-Transformer in MANSY employs a simple averaging strategy to combine the prediction results from different input-output heads. In other words, each head is treated equally and assigned with the same weight for combining prediction. To further improve the prediction accuracy of the model, one potential approach is to introduce an adaptive weighting scheme. To be specific, each head is initially assigned with the same weight. As the model makes prediction over time, the assigned weights can be dynamically adjusted based on the historical accuracy of each head. The underlying rationale is that a head with higher accuracy indicates that it captures user's viewing pattern more accurately, and thus should be allocated with a higher weight.

As for bitrate selection, MANSY currently focuses on optimizing the streaming services when users' QoE preferences

5. Recall that in Section 2.2, we also train *RLVA* with mulitple QoE preferences and it suffers from the similar catastrophic forgetting and generalization problems.

are known in advance. To unleash its full potential, MANSY can be complemented with the existing active research on inferring users' QoE preferences in practice [16] [44]. One viable approach is crowdsourcing [45], where users are invited to participate in questionnaires to gather information about their specific video watching preferences (e.g., priority of video quality). Alternatively, users' QoE preferences can also be inferred from their historical video watching behaviors (e.g., manual bitrate switching frequency) [16] [46]. For example, Li et al. [16] construct a behavior dataset by collecting users' behaviors of video watching. During the online service phase, users' behaviors will be collected as they watch the videos, and a Bayes' theorem based method is utilized to calculate their QoE preferences based on the collected behaviors and constructed dataset. The above approaches can be integrated into MANSY to efficiently infer users' preferences to optimize bitrate selection.

9 Conclusion

In this paper, we propose MANSY, a novel tile-based immersive video streaming system that fully captures user diversity to improve generalization. MANSY incorporates a Transformer-based viewport prediction model with an efficient Multi-viewport Trajectory Input Output (MTIO) architecture to reduce the prediction bias, so that it can generalize across users with diverse viewing patterns. For bitrate selection, to accommodate users' diverse QoE preferences, MANSY leverages representation learning (RepL) to encourage the DRL agent to learn useful representations of users' preferences by augmenting the reward function with mutual information. Considering the difficulty of exact computation of mutual information, it designs an efficient NN model called QoE identifier to estimate mutual information for reward calculation. Extensive experiments with real-world datasets confirm the superiority of MANSY in viewport prediction and bitrate selection on both trained and unseen viewing patterns and QoE preferences.

References

[1] Thomas Alsop, "VR headset unit sales worldwide 2019-2024," https://www.statista.com/statistics/677096/vr-headsets-worldwide/, 2022.

[2] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *Proceedings of the ACM MobiCom*, 2018, pp. 99–114.

[3] Z. Ye, Q. Li, X. Ma, D. Zhao, Y. Jiang, L. Ma, B. Yi, and G.-M. Muntean, "VRCT: A viewport reconstruction-based 360° video caching solution for tile-adaptive streaming," *IEEE Transactions on Broadcasting*, vol. 69, no. 3, pp. 691–703, 2023.

[4] N. Kan, J. Zou, C. Li, W. Dai, and H. Xiong, "RAPT360: Reinforcement learning-based rate adaptation for 360-degree video streaming with adaptive prediction and tiling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 3, pp. 1607–1623, 2022.

[5] J. Tu, C. Chen, Z. Yang, M. Li, Q. Xu, and X. Guan, "PSTile: Perception-sensitivity-based 360° tiled video streaming for industrial surveillance," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 9, pp. 9777–9789, 2023.

[6] Q. Guimard, L. Sassatelli, F. Marchetti, F. Becattini, L. Seidenari, and A. D. Bimbo, "Deep variational learning for multiple trajectory prediction of 360° head movements," in *Proceedings of the ACM MMSys*, 2022, pp. 12–26.

[7] M. F. R. Rondon, L. Sassatelli, R. Aparicio-Pardo, and F. Precioso, "TRACK: A new method from a re-examination of deep architectures for head motion prediction in 360° videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5681–5699, 2021.

[8] Y. Zhang, Y. Guan, K. Bian, Y. Liu, H. Tuo, L. Song, and X. Li, "EPASS360: QoE-aware 360-degree video streaming over mobile devices," *IEEE Transactions on Mobile Computing*, vol. 20, no. 7, pp. 2338–2353, 2020.

[9] Y. Hu, Y. Liu, and Y. Wang, "VAS360: QoE-driven viewport adaptive streaming for 360 video," in *Proceedings of the IEEE ICMEW*, 2019, pp. 324–329.

[10] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "DRL360: 360-degree video streaming with deep reinforcement learning," in *Proceedings of the IEEE INFOCOM*, 2019, pp. 1252–1260.

[11] Z. Jiang, X. Zhang, Y. Xu, Z. Ma, J. Sun, and Y. Zhang, "Reinforcement learning based rate adaptation for 360-degree video streaming," *IEEE Transactions on Broadcasting*, vol. 67, no. 2, pp. 409–423, 2021.

[12] Y. Jin, J. Liu, F. Wang, and S. Cui, "Where are you looking? A large-scale dataset of head and gaze behavior for 360-degree videos and a pilot study," in *Proceedings of the ACM Multimedia*, 2022, pp. 1025–1034.

[13] Y. Lu, Y. Zhu, and Z. Wang, "Personalized 360-degree video streaming: A meta-learning approach," in *Proceedings of the ACM Multimedia*, 2022, pp. 3143–3151.

[14] Z. Li, P. Zhong, J. Huang, F. Gao, and J. Wang, "Achieving QoE fairness in bitrate allocation of 360° video streaming," *IEEE Transactions on Multimedia*, pp. 1–11, 2023.

[15] F. Wang, C. Zhang, J. Liu, Y. Zhu, H. Pang, L. Sun *et al.*, "Intelligent edge-assisted crowdcast with deep reinforcement learning for personalized QoE," in *Proceedings of the IEEE INFOCOM*, 2019, pp. 910–918.

[16] W. Li, J. Huang, S. Wang, C. Wu, S. Liu, and J. Wang, "An apprenticeship learning approach for adaptive video streaming based on chunk quality and user preference," *IEEE Transactions on Multimedia*, 2022.

[17] X. Wang, Y. Enokibori, T. Hirayama, K. Hara, and K. Mase, "User group based viewpoint recommendation using user attributes for multiview videos," in *Proceedings of the Workshop on Multimodal Understanding of Social, Affective and Subjective Attributes*, 2017, pp. 3–9.

[18] C. Wu, Z. Wang, and L. Sun, "PAAS: A preference-aware deep reinforcement learning approach for 360 video streaming," in *Proceedings of the ACM NOSSDAV*, 2021, pp. 34–41.

[19] X. Yao, T. Huang, C. Wu, R.-X. Zhang, and L. Sun, "Adversarial feature alignment: Avoid catastrophic forgetting in incremental task lifelong learning," *Neural Computation*, vol. 31, no. 11, pp. 2266–2291, 2019.

[20] M. A. Ganaie, M. Hu, A. Malik, M. Tanveer, and P. Suganthan, "Ensemble deep learning: A review," *Engineering Applications of Artificial Intelligence*, vol. 115, p. 105151, 2022.

[21] M. Havasi, R. Jenatton, S. Fort, J. Z. Liu, J. Snoek, B. Lakshminarayanan, A. M. Dai, and D. Tran, "Training independent subnetworks for robust prediction," in *Proceedings of the ICLR*, 2021.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in NeurIPS*, vol. 30, 2017.

[23] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[24] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in VR spherical video streaming," in *Proceedings of the ACM MMSys*, 2017, p. 193–198.

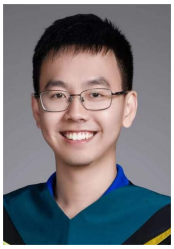
[25] M. O. Turkoglu, A. Becker, H. A. Gündüz, M. Rezaei, B. Bischl, R. C. Daudt, S. D'Aronco, J. D. Wegner, and K. Schindler, "FiLM-Ensemble: Probabilistic deep learning via feature-wise linear modulation," in *Advances in NeurIPS*, 2022.

[26] J. Chen, Z. Luo, Z. Wang, M. Hu, and D. Wu, "Live360: Viewport-aware transmission optimization in live 360-degree video streaming," *IEEE Transactions on Broadcasting*, vol. 69, no. 1, pp. 85–96, 2023.

[27] Y. Jin, J. Liu, F. Wang, and S. Cui, "Ebublio: Edge-assisted multiuser 360° video streaming," *IEEE Internet of Things Journal*, vol. 10, no. 17, pp. 15 408–15 419, 2023.

[28] L. Chopra, S. Chakraborty, A. Mondal, and S. Chakraborty, "PARIMA: Viewport adaptive 360-degree video streaming," in *Proceedings of the ACM WWW*, 2021, pp. 2379–2391.
[29] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI*, vol. 35, no. 12, 2021, pp. 11 106–11 115.
[30] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," *Advances in NeurIPS*, vol. 29, 2016.
[31] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, and D. Hjelm, "Mutual information neural estimation," in *Proceedings of the ICML*. PMLR, 2018, pp. 531–540.
[32] Y. Li, J. Song, and S. Ermon, "InfoGail: Interpretable imitation learning from visual demonstrations," *Advances in NeurIPS*, vol. 30, 2017.
[33] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," *arXiv preprint arXiv:1808.06670*, 2018.
[34] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE CVPR*, 2016, pp. 770–778.
[36] A. Nguyen and Z. Yan, "A saliency dataset for 360-degree videos," in *Proceedings of the ACM MMSys*, 2019, pp. 279–284.
[37] J. Van Der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck, "HTTP/2-based adaptive streaming of HEVC video over 4G/LTE networks," *IEEE Communications Letters*, vol. 20, no. 11, pp. 2177–2180, 2016.
[38] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360ProbDASH: Improving QoE of 360 video streaming using tile-based http adaptive streaming," in *Proceedings of the ACM Multimedia*, 2017, pp. 315–323.
[39] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proceedings of the ACM SIGCOMM*, 2017, pp. 197–210.
[40] W. Nguyen, "What is fov (field of view) in vr?" https://vrheaven.io/what-is-fov/#VR_Headsets_FOV_Comparison, 2024, accessed: 2024-03-19.
[41] VRCompare, "Meta quest pro," https://vr-compare.com/headset/metaquestpro, 2024, accessed: 2024-09-11.
[42] Z. Xia, Y. Zhou, F. Y. Yan, and J. Jiang, "Genet: Automatic curriculum generation for learning adaptation in networking," in *Proceedings of the ACM SIGCOMM*, 2022, pp. 397–413.
[43] J. Li, L. Han, C. Zhang, Q. Li, and Z. Liu, "Spherical convolution empowered viewport prediction in 360 video multicast with limited FoV feedback," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 19, no. 1, pp. 1–23, 2023.
[44] X. Zhang, Y. Ou, S. Sen, and J. Jiang, "SENSEI: Aligning video streaming quality with dynamic user sensitivity," in *Proceedings of the USENIX NSDI*, 2021, pp. 303–320.
[45] X. Zhang, H. Li, P. Schmitt, M. Chetty, N. Feamster, and J. Jiang, "VidPlat: A tool for fast crowdsourcing of quality-of-experience measurements," *arXiv preprint arXiv:2311.06698*, 2023.
[46] G. Gao, H. Zhang, H. Hu, Y. Wen, J. Cai, C. Luo, and W. Zeng, "Optimizing quality of experience for adaptive bitrate streaming via viewer interest inference," *IEEE Transactions on Multimedia*, vol. 20, no. 12, pp. 3399–3413, 2018.

Duo Wu is currently a Ph.D. student at Tsinghua University. He received his M.Phil. degree and B.Eng. degree from The Chinese University of Hong Kong, Shenzhen in 2024 and Jinan University in 2022, respectively. He has broad research interests in multimedia networking, reinforcement learning and large language models. He has published several first-author papers at top journals and conferences, including IEEE Transactions on Mobile Computing and ACM SIGCOMM.



Panlong Wu received the B.Eng. degree from the Department of Electrical and Electronic Engineering, Southern University of Science and Technology in 2022. He is currently pursuing the Ph.D. degree in the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. His current research interests include federated learning, foundation models and multimedia networking.

Miao Zhang (Student Member, IEEE) received her B.Eng. degree from Sichuan University in 2015, and her M.Eng. degree from Tsinghua University in 2018. She is currently a Ph.D. student at Simon Fraser University, British Columbia, Canada. Her research areas include cloud and edge computing, and multimedia systems and applications.

Fangxin Wang (S'15-M'20) is an assistant professor at The Chinese University of Hong Kong, Shenzhen (CUHKSZ). He received his Ph.D., M.Eng., and B.Eng. degree all in Computer Science and Technology from Simon Fraser University, Tsinghua University, and Beijing University of Posts and Telecommunications, respectively. Before joining CUHKSZ, he was a postdoctoral fellow at the University of British Columbia. Dr. Wang's research interests include Multimedia Systems and Applications, Cloud and Edge Computing, Deep Learning, and Distributed Networking and System. He leads the intelligent networking and multimedia lab (INML) at CUHKSZ. He has published more than 50 papers at top journal and conference, including INFOCOM, Multimedia, VR, ToN, TMC, IOTJ, etc. He was selected in the 8th Young Elite Scientist Sponsorship Program, CUHKSZ Presidential Young Scholar, and a recipient of SFU Dean's Convocation Medal for Academic Excellence. He serves as an associate editor of IEEE Transactions on Mobile Computing, TPC chair of IEEE Satellite 2023, TPC member of IWQoS, ICC, BigCom and reviewer of many top conference and journals, including INFOCOM, ToN, TMC, JSAC, etc.