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

Paper ID: MM-014184

**Author:** Xingyan Chen, Mu Wang, Changqiao Xu, Yu Zhao, Lujie Zhong, Gabriel-Miro Muntean

We thank sincerely the anonymous reviewers for their very constructive comments. Their suggestions helped us to improve the quality of this manuscript. We would also like to thank the editor for coordinating the review of this paper. Detailed responses on how the reviewers’ comments are addressed in this paper revision follow.

**Responses to comments from Editor**

**Dear Editor**

We appreciate your and reviewers time spent on our submitted manuscript and valuable suggestions made. Based on your comments and those of reviewers, we revised the previous version of our manuscript in the following main directions:

1. In section 3, we have edited the MEC-DC framework presentation providing more details about the framework in relation to the cooperative processing, buffer evolution and playback freeze clearer.
2. In section 5, we have improved the description of how our algorithms works with tiled content and we have clarified the adaptive bitrate streaming process.
3. In section 6, we have provided details about the prototype-based experimental setup and we have made the description clearer.
4. Finally, we have carefully checked and fixed all grammatical mistakes found in this manuscript. The references have been updated accordingly.

To improve the readability of the response, we have used blue color to highlight the revised/newly added parts. Additionally, to help differentiate the references from the manuscript and those in this response to reviewer comments document, we have labeled the references from the manuscript by [#], and the references from this response letter by [Reference #]. The detailed responses to the reviewers’ comments are as follows. Many thanks again for your efforts put when reviewing our manuscript.

**Responses to Reviewer 1**

**Comment 1:**

*The biggest point for me is about the potential real application of the solution. In normal AR/VR or other similar applications, such a system will require many processing power for functioning. It would be interesting to see the system working on real-time applications. At least, a major discussion on this point should be provided by the authors.*

**Answer:**

Thanks for the reviewer’s insightful comment. VR video processing contains multiple complex and computing-intensive tasks such as capturing, tilling, transcoding, rendering. This makes the live VR video streaming costly and difficult to be applied in real scale at present. In addition, the training process of reinforcement learning is also a computing-intensive task. It is fair to say that the limitation of computing resources is not only one of the main reasons that hinder the development of VR services, but also the challenge of the deployment of our solution in real-time applications.

To reduce the computational cost of online model training, we reduce the number of training iterations and the number of neural network layers of actor and critic networks. The model training time is reduced, and the supplementary experimental results are shown as in figure X. In addition, we further test the time consumption of our solution in prediction and training processioning. In our updated manuscript, **we have expanded the description of the system deployment for live VR video streaming and provided more experimental results to verify the feasibility of our solution on real-time applications in section VI.**

The federated transmission framework also has many advantages. For example, this design prevents data transmission delay for viewers’ viewing records over the networks. This is because if we deploy the model training on the server side, the servers need to aggregate a large number of viewers’ data for processing. Second, immersive livecast service has stringent latency requirements (i.e. reaction within 20ms and at least 60Hz refresh rate) to avoid motion sickness [3]. This further requires us to deploy the model training task at the network edge. Third, the federated solution reduces the risk of viewer privacy leakage. Viewers do not need to upload their private personal viewing traces to the server and instead upload the local prediction models or gradient loss in each epoch. Fourth, leveraging viewers’ devices to train neural network models or perform video transcoding is a promising alternative. Several related studies [Reference 1]-[Reference 4] have used crowd computing paradigm which leverages the viewers’ devices to perform computing tasks. For example, the authors in [Reference 1] mentioned that “As we observed, the viewer base in major Crowd Livecast Services (CLS) systems is always much larger than the channel base (the number of total channels) at any moment, indicating the existence of a huge amount of potential computational resources.” In [Reference 4], the authors stated that “We witness huge computational resources among the massive fellow viewers that could potentially be used for transcoding.” Meanwhile, the Cisco Annual Internet Report [Reference 5] revealed that combining device capabilities with faster, higher bandwidth, and more intelligent networks will facilitate broad experimentation and adoption of advanced multimedia applications. Each year, various new devices in different form factors with increased capabilities and intelligence are introduced and adopted in the market. Hence, many viewers’ devices-assisted computing paradigms have been proposed, such as Crowdsource-based [Reference 6] solution, Mobile Device Cloud [Reference 7], Distributed Computing Network [Reference 8], and Dispersed Computing [Reference 9]. It can be seen that leveraging edge resources such as viewers’ devices for computing tasks is one of the potential solutions to achieve high-quality panoramic livecast services in the future.

**The content in the revised version of the paper reads as follows:**

1. M. Ma, et al., "Characterizing user behaviors in mobile personal livecast: Towards an edge computing-assisted paradigm," ACM Trans. Multimedia Comput. Commun. Appl., vol. 14, no. 3s, 2018.
2. H. Pang, et al., "Optimizing personalized interaction experience in crowd-interactive livecast: A cloud-edge approach," in Proceedings of the 26th ACM International Conference on Multimedia, 2018, pp. 1217–1225.
3. R.-X. Zhang, et al., "Livesmart: A qos-guaranteed cost-minimum framework of viewer scheduling for crowdsourced live streaming," in Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 420–428.
4. R.-X. Zhang, et al., "A practical learning-based approach for viewer scheduling in the crowdsourced live streaming, " ACM Trans. Multimedia Comput. Commun. Appl., vol. 16, no. 2s, 2020.
5. Cisco Annual Internet Report (2018–2023) White Paper, March 9, 2020. <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>.
6. J. Ren, et al., "Exploiting mobile crowdsourcing for pervasive cloud services: challenges and solutions," in IEEE Communications Magazine, vol. 53, no. 3, pp. 98-105, March 2015.
7. V. Balasubramanian, et al., "Exploring Computing at the Edge: A Multi-Interface System Architecture Enabled Mobile Device Cloud," 2018 IEEE 7th International Conference on Cloud Networking (CloudNet), Tokyo, 2018, pp. 1-4.
8. J. Zhang, et al., "Optimal Control of Distributed Computing Networks with Mixed-Cast Traffic Flows," IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, Honolulu, HI, 2018, pp. 1880-1888.
9. M. R. Schurgot, et al., "A Dispersed Computing Architecture for Resource-Centric Computation and Communication," in IEEE Communications Magazine, vol. 57, no. 7, pp. 13-19, July 2019.

**Comment 2:**

*Some discussion on coding and the overload of coding is lacking. What about coding with adaptive streaming over HTTP?. Is it realistic to include federation when we have coding and streaming protocols?*

**Answer:**

We thank the reviewer for this valuable comment and are sorry for the lack of discussion on coding and the overload of coding. Our main consideration is to save bandwidth resources and achieve better VR service performance by providing high-accuracy FoV prediction. The FoV prediction model achieves bandwidth saving and service improvement by guiding the tile-level 360-degree video streaming. In this process, the FoV prediction model predicts the viewers’ region of interest. Based on the prediction results, the content servers split the omnidirectional image into multiple tiles and transcode each tile into multiple standard resolutions. The servers select tiles with appropriate resolution according to the network condition and steam them to the users. Since the main contribution of our solution is focused on the FoV prediction and content delivery in the multiple users system, the discussion of video coding and adaptive streaming is lacking. To this end, we revised the manuscript to clarify the two aspects as follows. First, we introduce the integration of our solution and adaptive streaming over HTTP. Secondly, a possible deployment scenario of the coding and streaming protocols under the federated framework is discussed.

**The content in the revised version of the paper reads as follows:**

**Reviewer: 2**

1. *It is unclear whether the system distributes content frame-by-frame or in segments (a certain number of frames in a segment or chunk), similar to DASH. Distributing tiles on a frame-by-frame basis does not seem very realistic. Also, what compression algorithm is used for the tiles? For a realistic system a modern codec should be used.*

**Answer:**

Thank you for this valuable comment. To support generality, we believe that “each video is divided into small video segments for transmission” which is stated in section III.B of the original manuscript. For example, if the AVC or SVC standards are considered, the unit of data transmission is one video segment. Setting the unit to the video frames is not a good choice in a real case, as it not only gives up the advantage of video coding (compression) but also increases the complexity of the rate control algorithm. To avoid ambiguity, we revised our manuscript clearly indicating that we refer to “video segments” in transmission phase.

In sections II,III.B and VI.A, we show that the equirectangular projection technique is adopted in our solution. Such as “We assume that the PLS system processes the sphere video with the equirectangular projection” in section III.B, “To simulate the tiled mechanism of panoramic video, we developed a visual playground for FoV prediction based on OpenAI-Gym6 library, which converts 360◦ videos into tile-based (5 tiles × 5 tiles) streaming with equirectangular projection” in section VI.A. The video after projection processing can be compressed by standard video coding technology. Currently, the most widely used video coding technologies are H.264, H.265, AVS2, etc. Under the premise of ensuring the same picture quality, the compression efficiency of H.265 and AVS2 is about 50% higher than that of H.264. The target compression efficiency of the next-generation coding technology H.266 and AVS3 is twice that of H.265 and AVS2. Besides, for 8K/50P video using H.265 or AVS2 encoding, the required bitrate is about 80-100Mbps.

In our manuscript, our main consideration is to provide high-accuracy FoV predictions. The FoV prediction requires frame-level analysis since the input of the prediction model is a sequence of the user viewing records for video frames. On the viewer side, the local FoV prediction model predict the viewer’s future FoV prediction results and selects the appropriate resolution for each tile according to the information provided by the video server, such as the MPD file of DASH. In general, the server produces video streaming for each tile specifically. Therefore, the viewers need to notify the server of the resolution for every portion of tiles they demand. The server then transmits the tile-level video steaming to the viewers in units of video segments. We believe “frame-level” data is needed for the prediction phase, while data transmission works at “segment-level”.

**The content in the revised version of the paper reads as follows:**

1. *What kind of tile quality is streamed during the distributed learning phase? During the learning some information may not be available. Does the system start with all low-quality tiles? How does the startup phase work?*

**Answer:**

We greatly appreciate the reviewer for this valuable comment and thank him/her for making us consider again the statement in section III.C. We modified this part and provided more details of how the FedLive framework works.

When new viewers join the system, the server delivers the unified model and low resolution tiled 360-degree video data to them. At this point, the viewers use the unified model to predict their FoV and pre-fetch the high resolution tiled video data from the server. Then, when enough viewing records are collected, the viewers train the unified model and update it to get their local models. Meanwhile, the gradient loss of model training is also uploaded to the server for the user cluster and upgrading of the unified model. On the server, the latest version of the unified model is distributed to all viewers periodically.

Since the transmission phase and the distributed learning phase occur simultaneously, the quality of the transmitted tiles depends on the predicted FoV and the network condition. We believe new viewers should start with low-resolution content in the startup phase. While the received unified model provides the FoV prediction results, the server will transmit high-resolution tiles to users.

We thank the reviewer again for pointing out this important aspect. To better describe the framework’s workflow, we reorganized the manuscript in Section III.C. The revised paper content is as follow:

1. *How the system deals with the time shift is not very clear. They state on page 4 that the time is slotted. Does that mean that clients will view the content at different slot numbers? Also, the delay from the origin to the various CDN servers can be variable, and then the delay from the CDN servers to the clients can be variable. It is not very clear how this is taken into account in the proposed formulation. Furthermore, if this is one of the distinguishing features from other techniques, then their should be some experiments presented that show how FedLive is handling these different delays well.*

**Answer:**

We thank the reviewer for this insightful comment. First, the authors agree with the reviewer's opinion about the time shift. We believe that different viewers view different images (or frames) of 360-degree live video streaming in asynchronous network systems since there is a time between viewers requesting and receiving the same video frame. This time shift phenomenon is inherent for the complex network system, which can not be ignored or eliminated. Therefore, the aim of our paper is not to solve this phenomenon but to design a matched solution, and we offer FedLive.

FedLive is a distributed asynchronous content delivery framework based on federated learning, that achieves multi-viewer cooperative training of the FoV prediction model, similar viewer clustering, and clustering-based collaborative content delivery. First, the cooperative training provides the unified model by integrating the gradient loss of different viewers. We believe the model combines the knowledge of different viewers since it is trained using the gradient loss of different viewers. This design matches the time shift phenomenon because the FoV movement of the preceding viewers can be used as pre-knowledge for the FoV prediction of the subsequent viewers. Second, clustering-based collaborative content delivery provides new opportunities for users to access data from other similar viewers. Since most viewers have similar viewing behaviors (based on the analysis of the real-world datasets) and the time shift phenomenon, the preceding viewer is a potential content provider for the subsequent viewers with similar behaviors. FedLive allow viewers to exchange data, which is one of the designs that supports the time shift phenomenon.

We carefully design the FedLive framework to adapt the time shift phenomenon. In the formulation, we mainly consider the optimization of FoV prediction and rate control. The formulations of the reward equation and the utility function are separate about viewers, thus ensuring that the algorithm can be designed to be distributed and asynchronous. In addition, we draw on the design idea of the asynchronous advantage actor-critic algorithm design, and propose a variational inference version of the asynchronous advantage actor-critic algorithm. Therefore, the distributed and asynchronous issues caused by the time shift phenomenon are also considered in problem formulation and algorithm design.

We thank the reviewer again for pointing out this important aspect. To better describe how FedLive works with the time shift phenomenon, we reorganized the manuscript in Section III.C, IV.A&C, and V.B. The revised paper content is as follow:

1. *Pages 3 and 4, Fig. 1(b): The caption in Fig. 1(b) and the text on page 4 says that this is a "heapmap". Usually this kind of graph with a color spectrum from blue (cold) to red (hot) is called a heatmap. Also, it would be good if in the figure caption it would be stated what the heat scale from 0 to about 48 represents. This figure should be explained better. Is tile 0 adjusted for the viewing direction, or is* *tile 0 always the same tile in the video? Also, if tile 0 is the viewing direction, then one would expect that the tiles immediately to the right and the left would experience higher visibility. But in this figure only tiles with positive indices show a high visibility. Why is that the case? Also, does tile 72 wrap around to tile 0?*

**Answer:**

We thank the reviewer for pointing out this issue. We have carefully checked the entire paper and fixed all grammar errors found. We have also clarified the description of the figure caption and modified the explanations about Fig. 1(b) to make it clearer in the manuscript.

Fig. 1(b) is generated by the rules provided in [35], and the saliency dataset can be found in [Reference 1]. In this dataset, A. Nguyen and Z. Yan partitioned the 360-degree video frame into 144(9\*16) pieces with equirectangular projection. Please see Diagram 1 for details.

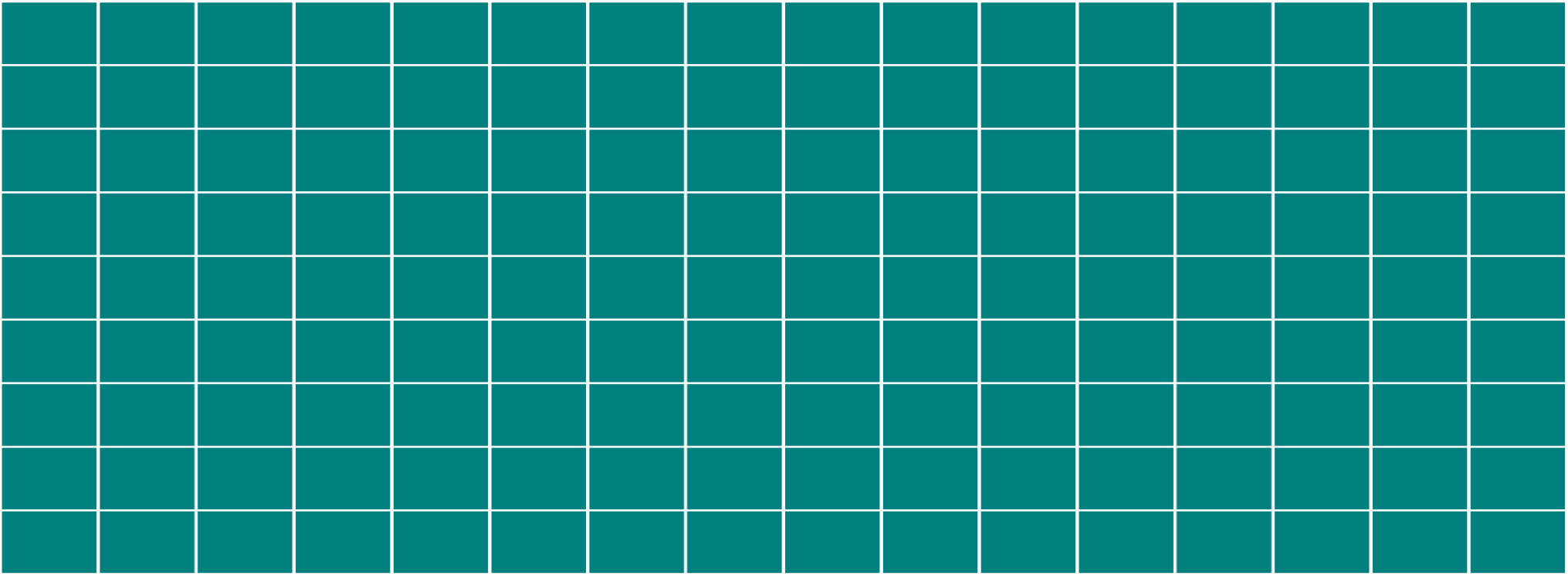


Diagram 1 Tiling mechanism of 360-degree video frame.

The “tile 0” represents the tile that is within FoV region of the greatest number of viewers, and it is neither the viewing direction nor always the index of the same tile in the video frames. In Fig. 1(b), we only show the “popularity” of the top 72 tiles and the X-axis represents the rank number of the tile rather than the index number of the tiles. Since the X-axis is the rank order, there are only positive indices. We set the popularity decreases with the rank number increasing which represents the rank number 0 is the most popular tile. Therefore, video tile ranked 0 does not refer to a specific index number of the tile, and the most popular tile may change in different video frame. The heatmap shows that most viewers are focused on the popular tiles at different video frames.

The revised content is shown as follows:

[1] https://zenodo.org/record/2641282#.YyPaYnZBw7d

1. *Page 9: "We achieved the time-shift phenomenon by configuring various startup delay for different viewers." If a viewer joins a stream with a delay, i.e., at a later point in time, then, if the viewer joins the latest point in the live stream (the so-called "live edge"), then it will be synchronous with other clients. Only if a client would start the stream later, but from the beginning of the stream, then there would be a time-shift. Please explain in more details how a client joins the live stream and how the time-shift is set in the experiments.*

**Answer:**

We thank the reviewer for this valuable comment. We agree with the comment and modified the description in this section to clarify the “*startup delay configuration*” and how it achieves the “*time-shift*”. In our experiment, “*startup delay*” means that the client starts the video streaming from the beginning of the video with a large startup delay.

The text has been revised accordingly:

1. *Page 9, Fig. 3: Only Edge servers 1 and 2 have network bandwidths labelled in Fig. 3: 100 Mbps. What are the capacities of the other links in the system? What bandwidth do the high quality tiles require and what bandwidth do the lower quality tiles require? It seems that with only 200 Mbps bandwidth from the server it would be nearly impossible to support 48 high-quality clients (that is only ~4 Mbps per client).*

**Answer:**

We thank the reviewer for this insightful comment. For real cases, each viewer should be equipped with an access point. Therefore, access points 1 and 2 represent two sets of access devices for viewers. First, we supplemented the capacities of the other links to Fig. 3. Second, we added a table to descript the bandwidth requirement for different resolutions in Section VI. In addition, we briefly explain how the simulation works in this network topology and add a brief introduction about that in section VI.

In the updated manuscript, we supplemented the bandwidth information from the access point to the viewers’ devices and added more details to Fig. 3. On the viewers’ side, each viewers’ device is configured to access the access point with 100Mbps link as follows.

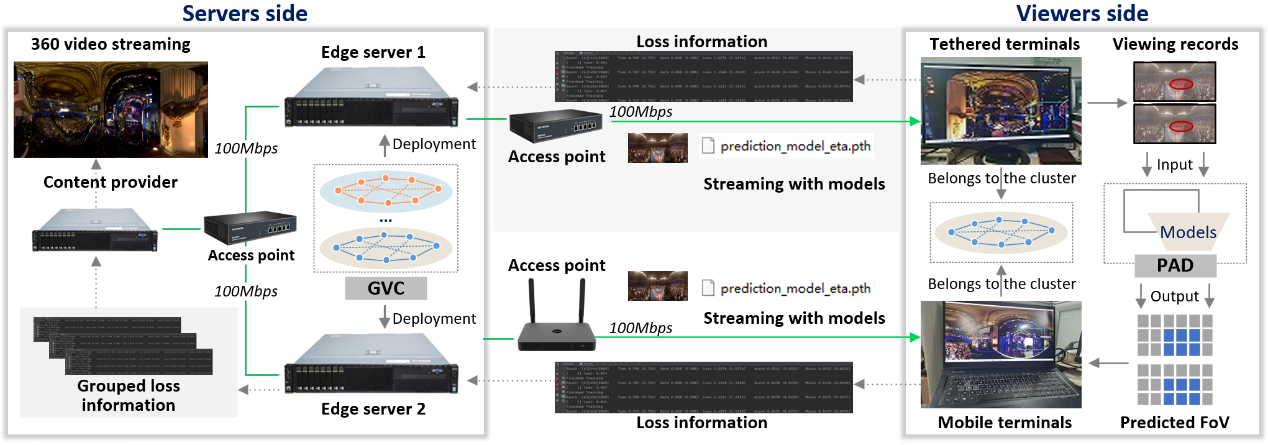
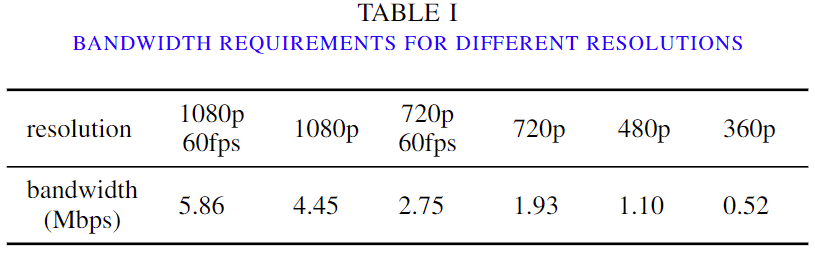


Fig. 3. The topology diagram of the prototype system.

The resolution of panoramic video is generally between 8K and 12K, and we split the 360-degree video frame into 5\*5 tiles. We consider the highest resolution of tile is 1080P and the lowest is 360P. Therefore, we set the required bandwidth of each tile streaming for different resolutions according to [1]. The bandwidth requirements for different resolutions are shown in the following table.



The content provider and two edge servers build a simple content delivery network. In this network structure, viewers access the edge servers rather than the content provider, while creating a connection between the viewers and the content provider is possible. The viewers fetch the 360-degree video from the edge servers for our setup. The links between the content provider and the edge servers are responsible for delivering the data generated by the content provider to the edge servers. The 100Mbps bandwidth link is sufficient in this process since only one 360-degree live video stream needs to be transmitted. To avoid confusion, we added a brief introduction about Fig. 3 to section VI in the updated manuscript.

**The content in the revised version of the paper reads as follows:**

1. H. Pang, et al., “Optimizing Personalized Interaction Experience in Crowd-Interactive Livecast: A Cloud-Edge Approach,” *2018 ACM Multimedia Conference on Multimedia Conference*, Seoul, Republic of Korea, Oct. 2018.
2. *Page 9: For the experimental evaluation, please explain in more details what the metrics mean: accuracy, precision and recall. Are these about the FoV prediction results? How were these metrics calculated? For example, does the FoV need to overlap 100% with the high quality tiles? Please describe how the results were calculated in more detail.*

**Answer:**

The authors thank the reviewer for this suggestion, which has helped us improve the performance evaluation. Following the reviewer recommendation, we revised the manuscript in two parts: the description of the key metric, such as “*accuracy*”, “*precision*”and“*recall*”, for the experience has been added to Section VI.B and additional details of experimental results were included to Section VI.C

The revised content is shown as follows:

1. *Page 11, Fig. 12: This figure also needs more explanation. What is the bandwidth of the different tiles? How many different tile qualities are there? How do you calculate the bandwidth savings? How do you calculate the bitrate change ratio?*

**Answer:**

We thank the reviewer for this valuable comment and very sorry for our loose description of the Fig. 12. To this end, we revised the manuscript in three parts: 1) We clarified the description of the four metrices; 2) We provided more details of the experiment settings. 3) We added some references to support the settings in our experiment.

The detail of revisions are as follows:

**Reviewer: 3**

1. *The authors clearly explain their problem. Instead, the explanation of the experimental part is not always clear. For instance, some of the parameters are not reported (e.g. the capacity of all the links involved in the scenario) Even the employed metrics, although already used by other previous work, should be defined*

**Answer:**

Thank you for this valuable comment. The authors agree with the reviewer's opinion and are very sorry for our loose description of some parameters and metrics. In our updated manuscript, we have revised the paper as follows. First, we have supplemented Table X in section 6, which shows the settings of all necessary experimental parameters. Next, we have revised section 6 and introduced the metrics used in performance evaluation. Finally, we have provided more details on the practical deployment and showed more performance comparisons with the state-of-the-art solutions. The text has been revised accordingly:

1. *The proposed approach is clear and technically sound but it is not clear how the system/algorithm would work/perform during the initial phase, when the distributed learning system is at work.*

**Answer:**

We greatly appreciate the reviewer for this valuable comment and we thank him/her for making us consider again the statement about the the algorithm design and experimental deployment. Thus, we added more details, such as initialize settings of our algorithm and the startup process, in our revised version of the manuscript in Section 5 & 6. The revised content is shown next:

1. *The main doubts emerging from reading the paper are related to the actual feasibility of the approach, especially related to the processing required and the compatibility with actually employed data tranmission protocols for the considered applications. This should be at least discussed.*

**Answer:**

The authors thank the reviewer for this insightful comment. We must admit that the prediction process based on reinforced variational inference technology requires intensive computing resources. According to prototype-level tests, the online learning process of our prediction model is time-consuming, which is one of the biggest challenges in the deployment and application. Therefore, the resource consumption of the local model training on the viewer side needs to be discussed. For this purpose, **we supplemented the experiment results and provided the feasibility analysis of our solution in section VI.**

Deploying model training on the viewer side has many advantages. For example, this design prevents data transmission delay for viewers’ viewing records over the networks. This is because if we deploy the model training on the server side, the servers need to aggregate a large number of viewers’ data for processing. Second, immersive livecast service has stringent latency requirements (i.e. reaction within 20ms and at least 60Hz refresh rate) to avoid motion sickness [3]. This further requires us to deploy the model training task at the network edge. Third, the federated solution reduces the risk of viewer privacy leakage. Viewers do not need to upload their private personal viewing traces to the server and instead upload the local prediction models or gradient loss in each epoch. Fourth, thanks to the rapid advancement in hardware, leveraging viewers’ devices to train neural network models or perform video transcoding is a promising alternative. Several related studies [Reference 1]-[Reference 4] have used the crowd computing paradigm. For example, the authors in [Reference 1] mentioned that “As we observed, the viewer base in major Crowd Livecast Services (CLS) systems is always much larger than the channel base (the number of total channels) at any moment, indicating the existence of a huge amount of potential computational resources.” In [Reference 4], the authors stated that “We witness huge computational resources among the massive fellow viewers that could potentially be used for transcoding.” Meanwhile, the Cisco Annual Internet Report [Reference 5] revealed that combining device capabilities with faster, higher bandwidth, and more intelligent networks will facilitate broad experimentation and adoption of advanced multimedia applications. Each year, various new devices in different form factors with increased capabilities and intelligence are introduced and adopted in the market. Hence, many devices-assisted computing paradigms have been proposed, such as Crowdsource-based [Reference 6] solution, Mobile Device Cloud [Reference 7], Distributed Computing Network [Reference 8], and Dispersed Computing [Reference 9]. It can be seen that leveraging edge resources such as viewers’ devices or edge servers for computing tasks is one of the potential solutions to achieve high-quality panoramic livecast services in the future.

**The content in the revised version of the paper reads as follows:**

1. M. Ma, et al., "Characterizing user behaviors in mobile personal livecast: Towards an edge computing-assisted paradigm," ACM Trans. Multimedia Comput. Commun. Appl., vol. 14, no. 3s, 2018.
2. H. Pang, et al., "Optimizing personalized interaction experience in crowd-interactive livecast: A cloud-edge approach," in Proceedings of the 26th ACM International Conference on Multimedia, 2018, pp. 1217–1225.
3. R.-X. Zhang, et al., "Livesmart: A qos-guaranteed cost-minimum framework of viewer scheduling for crowdsourced live streaming," in Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 420–428.
4. R.-X. Zhang, et al., "A practical learning-based approach for viewer scheduling in the crowdsourced live streaming, " ACM Trans. Multimedia Comput. Commun. Appl., vol. 16, no. 2s, 2020.
5. Cisco Annual Internet Report (2018–2023) White Paper, March 9, 2020. <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>.
6. J. Ren, et al., "Exploiting mobile crowdsourcing for pervasive cloud services: challenges and solutions," in IEEE Communications Magazine, vol. 53, no. 3, pp. 98-105, March 2015.
7. V. Balasubramanian, et al., "Exploring Computing at the Edge: A Multi-Interface System Architecture Enabled Mobile Device Cloud," 2018 IEEE 7th International Conference on Cloud Networking (CloudNet), Tokyo, 2018, pp. 1-4.
8. J. Zhang, et al., "Optimal Control of Distributed Computing Networks with Mixed-Cast Traffic Flows," IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, Honolulu, HI, 2018, pp. 1880-1888.
9. M. R. Schurgot, et al., "A Dispersed Computing Architecture for Resource-Centric Computation and Communication," in IEEE Communications Magazine, vol. 57, no. 7, pp. 13-19, July 2019.
10. *Finally, please proof read the paper to eliminate the few light typos still present. E.g., "heatmap" instead of "heapmap".*

**Answer:**

Many thanks for this valuable comment. We have carefully checked the entire paper and fixed all grammar errors found.